

GUIDANCE AND CONTROL OF TACTICAL MISSILES

Thomas Alan Grote

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

GUIDANCE AND CONTROL OF TACTICAL MISSILES

by

Thomas Alan Grote

December 1979

Thesis Advisor:

D. J. Collins

Approved for public release; distribution unlimited

T196176

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Guidance and Control of Tactical Missiles		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; (December 1979)
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Thomas Alan Grote		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE December 1979
		13. NUMBER OF PAGES 173
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) MOD6DF - STM		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This thesis discusses the conversion of the MOD6DF computer program for use on the IBM-360 computer at the Naval Postgraduate School. The functioning program was modified to investigate the impact miss distance for the Supersonic Tactical Missile. When the initial y-displacement error exceeded 1800 feet, the missile did not acquire the target. All errors smaller than this resulted in miss distances within 0.5 feet of the target. The midcourse guidance reference altitude was changed to reflect a sea-skimming missile. This		

simulation ran and impact was recorded. An attempt at adding random noise to the homing seeker was tried, but revealed that more information is required on this topic. The MOD6DF computer program was successfully converted and altered to run using the simplified ramjet model.

Approved for public release; distribution unlimited

Guidance and Control of Tactical Missiles

by

Thomas Alan Grote
Lieutenant, United States Navy
B.S.A.E., United States Naval Academy, 1974

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE

from

NAVAL POSTGRADUATE SCHOOL
December 1979

Thesis
G-86075
c.1

Approved for public release; distribution unlimited

Statement and Control of Foreign Assets

1

Thesis Title Page

Statement, Control of Foreign Assets

Statement, Control of Foreign Assets, 1970

Submitted to Federal Government of the
Republic of the United States of America

ABSTRACT

This thesis discusses the conversion of the MOD6DF computer program for use on the IEM-360 computer at the Naval Postgraduate School. The functioning program was modified to investigate the impact miss distance for the Supersonic Tactical Missile. When the initial y-displacement error exceeded 1800 feet, the missile did not acquire the target. All errors smaller than this resulted in miss distances within 0.5 feet of the target. The midcourse guidance reference altitude was changed to reflect a sea-skimming missile. This simulation ran and impact was recorded. An attempt at adding random noise to the homing seeker was tried, but revealed that more information is required on this topic. The MOD6DF computer program was successfully converted and altered to run using the simplified ramjet model.

TABLE OF CONTENTS

LIST OF TABLES -----	6
LIST OF FIGURES -----	7
TABLE OF SYMBOLS AND ABBREVIATIONS -----	8
ACKNOWLEDGEMENT -----	12
I. INTRODUCTION -----	13
II. MISSION REQUIREMENTS -----	14
A. SEPARATION -----	15
B. MIDCOURSE GUIDANCE -----	18
C. TERMINAL DIVE -----	18
D. TERMINAL GUIDANCE -----	19
III. COMPUTER SIMULATIONS -----	20
A. EXECUTIVE PROGRAMS -----	21
B. OPERATIONAL SUBROUTINES -----	23
C. MODULES -----	24
D. INPUT DATA DECK -----	29
E. AUXILIARY DECK -----	34
F. SEQUENCING -----	36
IV. COMPUTER PROGRAM CHECKOUT -----	40
A. MIDCOURSE GUIDANCE FLIGHT -----	41
B. TERMINAL GUIDANCE FLIGHT -----	47
V. TERMINAL FLIGHT DISTURBANCES -----	56
VI. CONCLUSION -----	64
COMPUTER OUTPUT -----	65
COMPUTER PROGRAM -----	68
LIST OF REFERENCES -----	171
INITIAL DISTRIBUTION LIST -----	173

LIST OF TABLES

I.	TXBA(073) Comparison (MIDCOURSE)	-----	48
II.	VXTP(286) Comparison (MIDCOURSE)	-----	48
III.	YTP(298) Comparison (MIDCOURSE)	-----	49
IV.	TXBA(073) Comparison (TERMINAL)	-----	49
V.	VXTP(286) Comparison (TERMINAL)	-----	50
VI.	YTP(298) Comparison (TERMINAL)	-----	50
VII.	Initial Conditions	-----	57
VIII.	Impact Results	-----	58

LIST OF FIGURES

1. Control Phases/Mission Phases Relationship -----	16
2. Flight Mode Sequencing -----	17
3. Typical Type 2 Card -----	32
4. Typical Type 3 Card -----	32
5. Module Diagram -----	38
6. Processing Sequence -----	39
7. Altitude vs. Time (MIDCOURSE) -----	42
8. Thrust vs. Time (MIDCOURSE) -----	44
9. VXTP vs. Time (MIDCOURSE) -----	45
10. YTP vs. Time (MIDCOURSE) -----	46
11. Altitude vs. Time (TERMINAL) -----	51
12. Thrust vs. Time (TERMINAL) -----	53
13. VXTP vs. Time (TERMINAL) -----	54
14. YTP vs. Time (TERMINAL) -----	55
15. Altitude vs. Time (TERMINAL-ZTP) -----	60
16. Impact Results -----	61
17. Altitude vs. Time (SEA-SKIMMER) -----	62

TABLE OF SYMBOLS AND ABBREVIATIONS

The following is a list of the abbreviations and some of the more common fortran symbols used in the MOD6DF computer program. Each symbol is defined in two ways. The primary source of identification is by its COMMON(3415) location and the second is by its fortran symbol. The fortran symbol is not always a good identifier since it will change from subroutine to subroutine (i.e. W and WE both are used for missile weight).

A. ABBREVIATIONS

<u>Symbol</u>	<u>Definition</u>
TP	tangent plane axes
BA	body axes
SA	stability axes

B. FORTRAN SYMBOLS

<u>Symbol</u>	<u>Definition</u>
TXBA(073)	missile thrust in the x-direction in BA (lb)
TYBA(074)	missile thrust in the y-direction in BA (lb)
TZBA(075)	missile thrust in the z-direction in BA (lb)
W(086)	missile weight (lb)
S(110)	missile reference area (ft ²)
CA(111)	drag coefficient
CY(113)	side-force coefficient
CZ(115)	normal force coefficient
CEAR(116)	mean aerodynamic chord (ft)
CMQ(118)	damping in pitch coefficient
CNR(119)	damping in yaw coefficient

<u>Symbol</u>	<u>Definition</u>
CLP(120)	damping in roll coefficient
CM(121)	pitching moment coefficient
CN(122)	yawing moment coefficient
CL(123)	rolling moment coefficient
CGI(136)	center of gravity (ft)
A(201)	moment-of-inertia about missile roll axis (x-body axis) (slug-ft ²)
B(202)	moment-of-inertia about missile pitch axis (y-body axis) (slug-ft ²)
CC(203)	moment-of-inertia about missile yaw axis (z-body axis) (slug-ft ²)
TSA(208)	angle between SA and EA (rad)
P(212)	missile angular velocity about x-BA (rad/s)
Q(216)	missile angular velocity about y-BA (rad/s)
R(220)	missile angular velocity about z-BA (rad/s)
AG(282)	unit conversion lb-slug
VXTP(286)	missile velocity in x-TP (ft/s)
XTP(290)	missile displacement in x-TP (ft)
VYTP(294)	missile velocity in y-TP (ft/s)
YTP(298)	missile displacement in y-TP (ft)
VZTP(302)	missile velocity in z-TP (ft/s)
ZTP(306)	missile displacement in z-TP (ft)
GZRO(404)	constant set to zero for flat earth gravitational field and set to one for a spherical gravitational field
ER(405)	angular velocity of earth (rad/s)
ALPO(406)	angle between north and x-TP (rad)

<u>Symbol</u>	<u>Definition</u>
OLAMO(407)	latitude origin of tangent plane
HO(414)	distance tangent plane is from earth (ft)
GO(415)	gravitational acceleration (ft/s^2)
HREF(501)	reference altitude for midcourse guidance (ft)
RE(503)	earth's radius (ft)
H(507)	altitude normal to earth (ft)
AMACH(520)	missile Mach number
THET(521)	missile pitch angle, TP (rad)
PSI(522)	missile yaw angle, TP (rad)
PHI(523)	missile roll angle, TP (rad)
GAMMAV(527)	vertical flight path angle, TP (rad)
VEL(528)	magnitude of missile velocity (ft/s)
VAT(529)	missile velocity (ft/s)
TF(550)	program termination time (s)
VAH(561)	computed speed of sound (ft/s)
ALAT(576)	latitude of target position (deg)
AZ(578)	azimuth of target position (deg)
DYNP(581)	dynamic pressure (psi)
FLGRJ(606)	constant set to zero indicates run will use ENGINE subroutine and when set to one run will use RAMJET subroutine (simplified ramjet model)
T(932)	actual time (s)
T1(933)	boost engine ignition (s)
T2(934)	commence acceleration command mode (s)
T3(935)	boost engine burn-out/port cover blow-in (s)
T4(936)	start ramjet engine (cruise) (s)
T5(937)	commence heading and altitude guidance control (s)

<u>Symbol</u>	<u>Definition</u>
T6(938)	commence terminal dive (s)
T7(939)	commence terminal guidance - search (s)
T8(940)	commence terminal guidance - track (s)
DMAX(1740)	maximum fin deflection (rad)
CPP(2669)	time between printouts (s)
ROLLO(2901)	initial roll angle (rad)
PITCHO(2902)	initial pitch angle (rad)
YAWO(2903)	initial yaw angle (rad)
STEP(2905)	determines executive program flow after staging
DOC(2909)	defines number of times COMMON will be printed
HMIN(2911)	minimum integration step size
HMAX(2912)	maximum integration step size
DER(1)(2913)	integration step size (s)

ACKNOWLEDGEMENT

This author is grateful to Professor D. L. Collins, NAVAL POSTGRADUATE SCHOOL, for his helpful discussions and suggestions. I also acknowledge NWC CHINA LAKE for their cooperation in obtaining the MOD6DF computer program.

I. INTRODUCTION

Research was undertaken to convert the MOD6DF computer program received from NWC China Lake for use on the IBM-360 computer at the Naval Postgraduate School. Once a functioning program was obtained, initial displacement error versus impact miss distance was investigated for the supersonic tactical missile. Additionally, the midcourse guidance reference altitude was changed to reflect a sea-skimming scenario and this was examined for its effect on the terminal guidance problem. This report not only discusses the aforementioned topics, but also describes the missile mission requirements and the MOD6DF computer program.

II. MISSION REQUIREMENTS

The Supersonic Tactical Missile (STM) mission is divided into six phases (Ref. 1).

- * initial conditions
- * separation
- * boost
- * transition
- * cruise
- * terminal

These divisions are based on the missile aerodynamics. The initial condition phase establishes the starting conditions for each launch. This is done while the missile is still attached to the launch platform. The separation phase starts when the missile is launched. The missile falls for approximately five seconds until the boost engine ignites. This initiates the boost phase which continues until the missile achieves Mach two. As the missile passes through Mach one, plume effects are encountered which the aerodynamics account for. At the end of the boost phase, the port covers blow in. This initiates the transition phase. This phase is very short and allows the debris to be ejected from inlet ports. The cruise phase commences when the ramjet engine ignites. This engine propels the missile until target impact. The terminal phase of the flight begins when the missile is commanded to dive from the cruise altitude. This phase concludes when the flight is terminated at target impact.

The six STM mission phases require four phases of control. These control phases are:

- * separation
- * midcourse guidance
- * terminal dive
- * terminal guidance

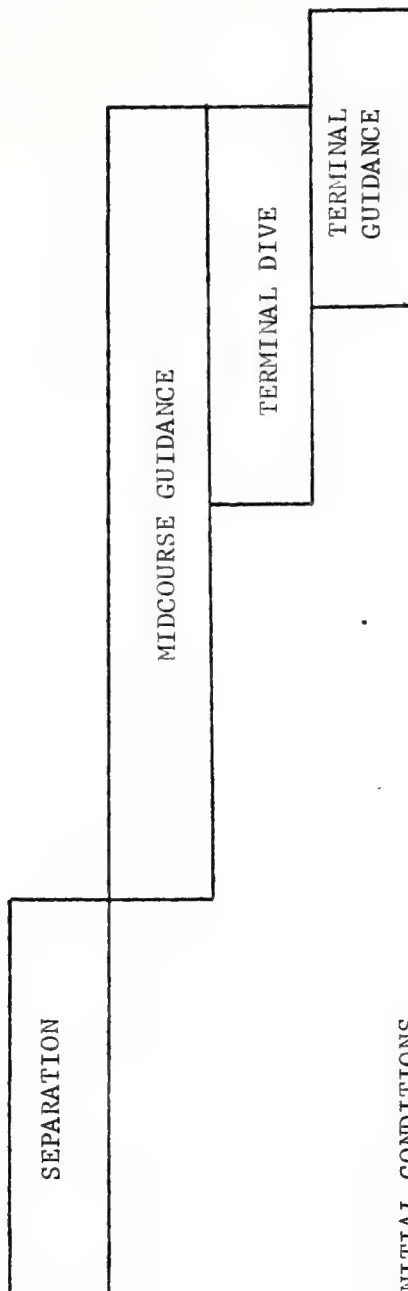
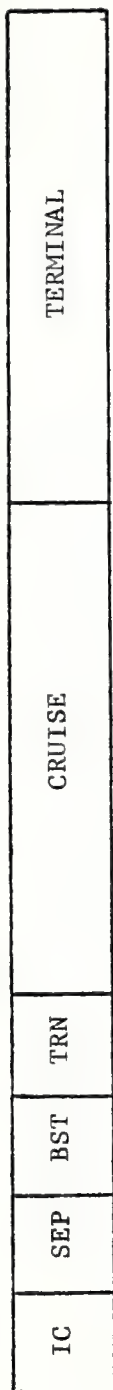
The four Guidance, Navigation, and Control (GN&C) system control phases are directly related to the mission phases. Figure 1 displays this relationship (Ref. 2). To obtain a successful flight, various types of control and guidance functions are required. Figure 2 (Ref. 3) illustrates the guidance control mode sequencing in relation to the mission phases and also indicates the critical missile switching times. The control phases are discussed below, along with the appropriate guidance modes.

A. SEPARATION

The separation phase commences upon launch. The missile is ejected downward from the launch platform. Additionally, the pitch attitude of the missile is commanded down. Since this portion of the flight is unguided, the ejection force and gravity are the only forces acting on the missile. At launch, the missile is required to be in the attitude command mode.

Five seconds into the flight, the missile pitch attitude is commanded up and the boost engine is ignited. The booster continues until the missile attains Mach two, at which time control is shifted from attitude to acceleration control mode. The acceleration control then requires the missile to maintain the normal and lateral accelerations at zero.

The last event to occur in the separation phase is inlet port cover blow-in. The time delay that allows the port covers to clear is a function of altitude.

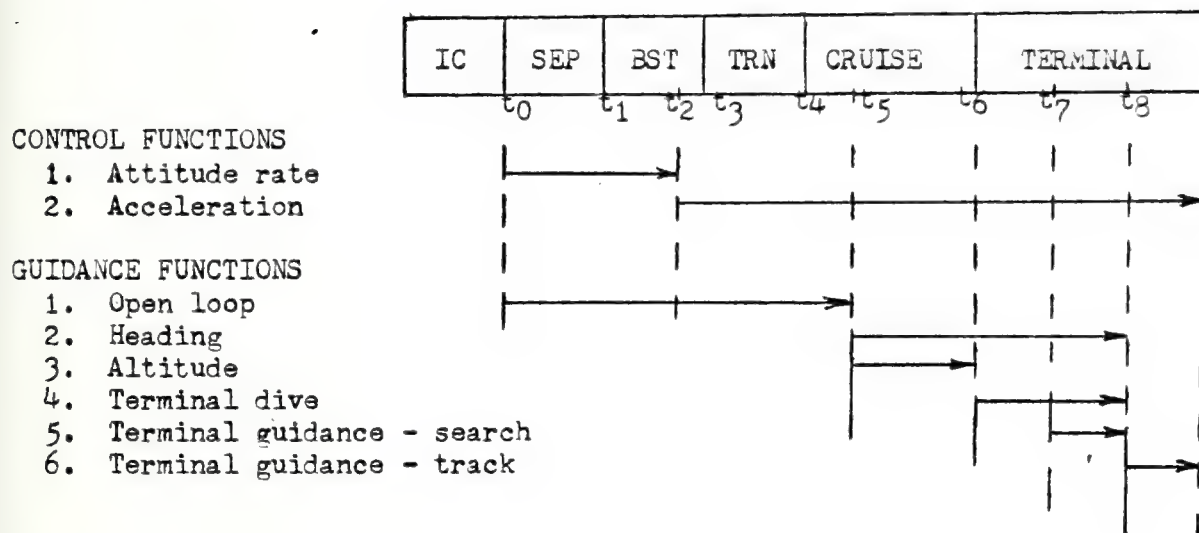


IC - INITIAL CONDITIONS
 SEP- SEPARATION
 BST- BOOST
 TRN- TRANSITION

CONTROL PHASES/MISSION PHASES
 RELATIONSHIP

FIGURE 1

- t₀ - LAUNCH: START SIMULATION
- t₁ - BOOSTER IGNITION
- t₂ - CONTROL SYSTEM MODE CHANGE FROM ATTITUDE TO ACCELERATION CONDITION AT M=2.0
- t₃ - PORT COVER BLOW-IN, BOOSTER BURNOUT
- t₄ - RAMJET IGNITION, t₃ + .3 sec
- t₅ - ENGAGE GUIDANCE MODES
- t₆ - DIVE COMMAND
- t₇ - SEEKER SEARCH MODE IS ACTIVATED
- t₈ - TERMINAL TRACKING



FLIGHT MODE SEQUENCING

FIGURE 2

B. MIDCOURSE GUIDANCE

Midcourse guidance begins upon completion of the separation phase. The moment the port covers are clear, the phases shift. During this phase two things are required. The GN&C must guide the missile to the established cruise altitude and then guide the missile (in the horizontal plane) to the predetermined target location.

To accomplish these requirements the GN&C system employs both altitude and heading control. The altitude control acts upon the pitch axis to drive the missile to the cruise altitude and then to maintain that altitude until the terminal dive phase commences.

The actual guidance work is performed by the yaw axis. The GN&C system uses the heading control to steer the missile to the target position. To generate the steering commands a guidance law is necessary. The guidance law should minimize the cross-track error and the final lateral acceleration. A minimum cross-track error allows for minimum flight time and minimum displacement error when target search is initiated. Minimizing the final lateral acceleration ensures that the seeker will continuously be pointing toward the target area. To insure accuracy, the guidance law must be able to perform these functions when subjected to disturbances such as wind and thrust misalignment.

C. TERMINAL DIVE

The terminal dive phase commences when the missile is commanded to dive. The actual time that this command is given is a function of the predetermined target coordinates. Upon diving, the missile must accelerate downward along a 60-degree (from horizontal) dive angle to the target position. Once the missile has steadied up in the proper dive aspect, terminal guidance commences.

D. TERMINAL GUIDANCE

During the terminal guidance phase the seeker is required to search, acquire, and track the target. The GN&C system then uses these tracking signals to direct the missile to the target. The seeker uses a preprogrammed search pattern to locate the target. When the target has been acquired, the seeker then commences tracking it and also notifies the GN&C system that the target is being tracked.

Target acquisition takes place when the target falls within the instantaneous four-degree beamwidth pattern as it traverses the scan pattern on the earth's surface. The computer program uses this assumption since the exact missile target acquisition mechanism has not yet been defined.

Upon acquisition, the GN&C system closes the seeker tracking loop. The radiometer error signal is defined as the difference between the look vector and the line-of-sight. This signal is used to reposition the antenna gimbals so the look vector and the line-of-sight coincide. Under the closed loop operation the radiometer output is proportional to the line-of-sight rate and this is the signal used to guide the missile to target impact.

III. COMPUTER SIMULATION

The modularized six-degree-of-freedom (MOD6DF) computer program was developed by the Litton Systems, Inc., Guidance and Control Systems Division (Ref. 4) to be used in analyzing missile guidance and control. The program uses a building block approach, where each module corresponds either to a missile subsystem or an environmental system. In its original form, the program is used primarily for terminal guidance of air-to-surface missiles. The Naval Weapons Center, China Lake modified the original program to specifically apply to the Supersonic Tactical Missile (STM). The modification also allows the user the capability of simulating any portion of the missile flight trajectory.

The MOD6DF computer program consists of four main decks and one auxiliary deck. The main decks include the executive programs, operational subroutines, modules, and input data deck. All the subprograms utilized in the integration and the sequencing of the modules are contained in the executive programs. The operational subroutines are used by the user to control the program while it is running. The physical system and the environmental subroutines are included in the modules. The input data deck contains all the information the executive programs need to execute the desired run. Lastly, the auxiliary deck consists of all those subprograms (subroutines and functions) that are required by the modules.

Once the user starts making simulations, he must concern himself with program sequencing. Proper sequencing is required to ensure a valid run is achieved. Since it is of such importance, sequencing will be discussed as the final part of this section.

A. EXECUTIVE PROGRAMS

The executive programs are the main core of the MOD6DF program. These subroutines have various functions in setting up, sequencing, integrating, and resetting the program. Since these functions are required by any type of analysis done, the user should never have to change any of these subroutines. Reference 5 should be consulted if the user desires more information about the content of any of the subroutines described herein.

1. Zero

Subroutine ZERO is used to set all the COMMON(3415) locations to the value zero. This is done to ensure that no erroneous information is used in the simulation.

2. Oinpt1

Subroutine OINPT1 is the basic input routine for the MOD6DF program. The normal input is from punched cards. However, inputs may also be read from tapes. The basic input cards will be discussed in the section covering the input data deck.

3. Auxi

Subroutine AUXI is used to call the initialization modules for each run. Additionally, it sets up the list of state variables which are used in AMRK.

4. Auxsub

Subroutine AUXSUB is used to call the dynamic modules. In calling the dynamic modules, AUXSUB sets and resets the lists needed for AMRK and the COMMON(3415) storage cells with the most recent values of the state variables and their derivatives.

5. Amrk

Subroutine AMRK is the integration subroutine. It uses a point-wise first order Runge-Kutta method. All the state variables to be integrated must be listed in COMMON(3415) and also must appear in a processing list.

6. Reset

Subroutine RESET is used to reinitialize up to fifty input parameters. This is done prior to the start of any repeated runs. To indicate which parameters are to be reset, the number one is punched in columns 46-60 on the respective type three card.

7. Return Group

The return group is a collection of all the unused modules. Since all the modules and their initialization modules are called by AUXI and AUXSUB, the unused ones must still remain in the deck. These are required to ensure proper linking when the computer attempts to link all the subroutines. All the subroutines in this grouping contain three cards. The three cards are the subroutine title, and a return and an end card.

8. Sub11, Sub12, Sub13

These subroutines are used to call the operational subroutines that are required. They call the routines in the order prescribed by the input data deck. The number at the end of each subroutine title indicates which operational subroutines it can call. For example, SUB11 can call STGE1.

B. OPERATIONAL SUBROUTINES

The operational subroutines provide the user with control of the program while it is running. The order in which the operational subroutines are called is specified by the input data deck. Since these routines assist the user in controlling the simulation, they can be reprogrammed. However, it is advised that they not be changed until the user has become quite familiar with the overall operation and sequencing of the MOD6DF program. For more in depth knowledge of the operational subroutines the user should consult Ref. 6, and the computer listing, which is contained at the end of this report.

1. Inpt1, Inpt2, Inpt3

These subroutines are available for new inputs during the simulated flight. The only one presently used in the program is INPT1. It is utilized to input a namelist file, which is used by ENGINE and is described there.

2. Oupt1, Oupt2, Oupt3

These subroutines allow for the print-out of up to fifty different variables during the flight simulation. OUP1 is not utilized in the deck. OUP2 is used when the desired output is to be put on tape. OUP3 is the basic output routine. It prints the desired output on regular computer paper.

3. Stge1, Stge2, Stge3

These subroutines allow for proper staging, run termination, etc.. Presently STGE1 is not being used. STGE2 is being used as the staging initialization subroutine. STGE3 is the primary subroutine of this group. It stages when impact with the earth is made, when the final time, TF(550), is reached, or when LCONV(2672) is set equal to two. All the tolerances for staging are listed in STGE3.

4. Cntr1, Cntr2, Cntr3

These subroutines allow the external dynamic control inputs to the modules. In the MOD6DF program these are not used.

5. Rndm1, Rndm2, Rndm3

These subroutines allow random noise to be added to the state variables generated in the modules. RNDM1 is not used in the program. RNDM2 is used as the initialization subroutine, while RNDM3 provides continuous noise values. These subroutines produce correlated noise values for as many modules as required. The noise values remain fixed during each individual integration cycle.

6. Auxa1, Auxa2, Auxa3 Auxb1, Auxb2, Auxb3 Auxc1, Auxc2, Auxc3

These subroutines are auxiliary routines that allow for external input, output, control, etc. of the modules. At the present time, none are utilized in the MOD6DF program.

C. MODULES

The modules are of prime importance to the user since they represent the 'model' of the dynamic system. In general, the model is described by ordinary non-linear time-varying differential equations with both random and deterministic forcing functions. The user must first reduce these equations to an equivalent system of first order equations, which can then be described by each module. Generally speaking, the physical system is so complex that this would be impossible to do. However, due to the modularity of the MOD6DF program, the user can think of each module as a completely independent system described by the equations within that module.

There are thirty-six possible modules divided into five functional

categories. Each group is identified by a letter which pertains to that groups function; A (airframe), C (computers), D (dynamics), G (geophysical), S (sensors). A complete printing of each module is contained in the computer program listing.

1. Airframe

a. Subroutine A1

This is the aerodynamic forces and moments modules. It calculates all the necessary forces and moments in body axes. These values are then used in the computation of the dynamics.

b. Subroutine A2

This is the missile aerodynamic coefficient module. It calculates the required coefficients using the information stored in the BLOCK DATA. Using the timing inputs, this routine computes the coefficients for the different effects. Some of the effects accounted for are; plume, separation, and control surfaces effectiveness. With this done the total coefficients are determined.

c. Subroutine A3

This is the missile propulsion module. The timing inputs are used to determine whether the missile is in free fall, boost, transition, or cruise phase. With this determined the correct engine subroutine (BOOST, RAMJET, ENGINE) can be called. Three variables are calculated using the body axes as the frame of reference. They are the missile thrust in all coordinate directions, the principal moments of inertia, and the missile weight.

d. Subroutine A4

This is the fin actuator module. The four control surfaces commands are calculated as either ideal actuators or as second-order

ones. In addition to control surfaces commands, the rate of change of yaw, and roll are computed.

e. Subroutine A5

This module is part of the return group.

2. Computers

a. Subroutine C1

This is the autopilot module for the STV-G. It uses the cruise engine ignition time to divide the routine into boost/separation and cruise phases. These two phases use different algorithms to calculate the turning moments in pitch/yaw and roll.

b. Subroutine C2

This is the guidance command module. Using the timing inputs, this routine is divided into separation/boost, dive/climb, cruise, terminal dive, and terminal homing sections. Each section uses slightly different algorithms to calculate the guidance commands to maintain the proper flight profile.

c. Subroutines C3 - C10

These modules are part of the return group.

3. Dynamics

a. Subroutine D1

This is the translational dynamics module. It computes the total acceleration in body axes and then converts them to the tangent plane reference. Then, accounting for aerodynamics, thrust, gravity, and coriolis, the velocity and acceleration are calculated.

b. Subroutine D2

This is the rotational dynamics module. With the principal axes as a reference, this subroutine computes the body angular rates and the attitude direction cosines.

c. Subroutines D3 - D5

These modules are part of the return group.

4. Geophysical

a. Subroutine G1

This is the gravitational and coriolis acceleration module.

It calculates the gravitational acceleration using one of two fields.

The user specifies the field to be used by an input card. To use a flat-earth gravitational field, GZRO(404) must equal 0.0, and to use a spherical gravitational field, GZRO(404) must equal 1.0.

b. Subroutine G2

This module is part of the return group.

c. Subroutine G3

This is the air data module. It computes the velocity, in all three coordinate directions, with respect to the air mass. These values are then resolved into body and stability axes. This module also computes all the properties of air by calling subroutine AIR. These values are stored in their COMMON(3415) locations for use in the other modules.

d. Subroutine G4

This module is part of the return group.

e. Subroutine G5

This is the coordinate conversion module. It takes the missile position, does a coordinate conversion and then it determines the position, velocity, and acceleration in the ECI system.

f. Subroutine G6

This module is part of the return group.

5. Sensors

a. Subroutine S1

This is the homing seeker module. It simulates the missile seeker and computes the seeker dynamics and Euler angle rates. Several flags are used to control the seeker sequencing:

- (1) FLAGS(335). FLAGS signals the start of the seeker search.
- (2) FLGT(317). FLGT signals when the seeker is locked-on.
- (3) FLGTS(347). FLGTS signals the end of search.
- (4) FLGD(336). FLGD signals when the seeker has detected the target.
- (5) FLAGLT. FLAGLT signals when the target is outside the seeker field of vision.

b. Subroutine S2

This is the radiometer module. It takes the target position and the ATIGS target position and converts them from body axes to the seeker axes. Using target position, the module then calculates the azimuth and elevation error signals.

c. Subroutines S3, S4

These modules are part of the return group.

d. Subroutine S5

This is the accelerometers and gyros modules. The user has the option of using ideal or digital accelerometers. To specify the type of accelerometer, the user must include the appropriate data statement in the subroutine. If ideal accelerometers are desired, FLGA must equal 0.0 and for digital accelerometers, FLGA must equal 1.0.

e. Subroutines S6 - S10

These modules are part of the return group.

D. INPUT DATA DECK

The input data deck provides the user with the means of specifying which operational subroutines and modules are to be utilized for the desired run. It also allows the user to set the starting conditions. In general, only the constants and the state variables must be given initial values. All quantities in COMMON not given initial values will be set to zero by ZERO. In addition to the state variables, the upper and lower error bounds must be initialized. There are seven types of input cards, each indicating a certain function.

<u>Type</u>	<u>Function</u>
0	read/write tape
1	operational subroutine to be called
2	module to be called
3	numerical input
4	printed output
5	parameter square and sum
6	termination and random noise generator input

A separate card is required for each subroutine, module, input, and output quantity. A sample computer printout of the input data deck is contained in the computer output section.

1. Type 0 Card

Type 0 cards are used to indicate if the type 3 inputs are to be read from or written onto an auxiliary tape. These procedures can be used rather than reading the inputs from a deck of cards. A typical card is defined by punching a zero in column 2. The field that covers columns 5 - 20 is used by the user for any descriptive statements with which he wishes to identify the input. Column 21 -25 contains the right-

justified integer number of the tape transport to be used. The number of the first record to be read is punched in column 31 - 45. The last field is column 46 - 60 which contains the number of records to be read. The last two fields may use either fixed or floating-point notation.

2. Type 1 Card

Type 1 cards are used to specify which operational subroutines are called during the flight simulation. This type card is identified by the number one in column 2. The second field is column 5 - 20 which contains any identifying information. This information is printed out when the data deck is read and allows the user to read exactly which subroutines were called. Column 21 - 25 contains the right-justified integer number which is the subroutine identifying number. The operational subroutine numbers are;

<u>Subroutine</u>	<u>Subroutine Number</u>
INPT1, INPT2, INPT3	2
OUPT1, OUPT2, OUPT3	3
STGE1, STGE2, STGE3	4
CNTR1, CNTR2, CNTR3	5
RNDM1, RNDM2, RNDM3	6
AUXA1, AUXA2, AUXA3	7
AUXB1, AUXB2, AUXB3	8
AUXC1, AUXC2, AUXC3	9

It should be noted that all or any of the subroutines listed under one number can be called by including only one card. The cards are placed in the data deck in the order in which they will be called. This order or sequencing will be explained further in section F, Sequencing. A typical type 2 card is identified by the number two in

column 2. In general, column 5 - 20 should contain the module title, but any pertinent information is allowed. The module number is punched in column 21 - 25 and it must be right-justified. A listing of the module numbers follows:

<u>Module</u>	<u>Module Number</u>	<u>Module</u>	<u>Module Number</u>
A1,A1I	2	D4,D4I	20
A2,A2I	3	D5,D5I	21
A3,A3I	4	G1,G1I	22
A4,A4I	5	G2,G2I	23
A5,A5I	6	G3,G3I	24
C1,C1I	7	G4,G4I	25
C2,C2I	8	G5,G5I	26
C3,C3I	9	G6,G6I	27
C4,C4I	10	S1,S1I	28
C5,C5I	11	S2,S2I	29
C6,C6I	12	S3,S3I	30
C7,C7I	13	S4,S4I	31
C8,C8I	14	S5,S5I	32
C9,C9I	15	S6,S6I	33
C10,C10I	16	S7,S7I	34
D1,D1I	17	S8,S8I	35
D2,D2I	18	S9,S9I	36
D3,D3I	19	S10,S10I	37

Note that either the module, the initialization module, or both may be called by including only one card in the deck. A sample of a typical type 2 card is shown in Figure 3 (Ref. 7).

COLUMN

2	5	20	21	25
2	MODULE 55			32

TYPICAL TYPE 2 CARD

FIGURE 3

2	5	20	21	25	31	45	46	60
3	WEIGHT			86				
					1152.0	---	1.0	---
					1.152	E+03	1.0	E+00

TYPICAL TYPE 3 CARD

FIGURE 4

4. Type 3 Card

Type 3 cards are used to set any COMMON(3415) location to any value other than zero. In general, four items must be initialized. The state variable initial values and any constants used in the flight simulation are the most obvious. Additionally, there are some constants associated with the executive programs and operational subroutines and the state variable upper and lower bounds which must be initialized. As with all cards, column 2 defines the type card and it must contain a three. Column 5 - 20 holds the statement describing the input. The user should be specific here since it will save him having to remember every COMMON(3415) location. The only other means of input identification is by column 21 - 25. These columns contain the right-justified COMMON location of the input. Columns 31 - 45 hold the actual numerical value of the input. The last field, column 46 - 60, contains the reset flag. If the reset flag equals one, the COMMON(3415) location and the numerical value are placed in the reset list. This list may contain up to fifty different variables. This, in the case of multiple runs, allows the variables to be reset to its initial value prior to each run without additional input cards. The sample card in Figure 4 (Ref. 8) shows that either fixed or floating-point notation may be used to input the numerical value and the reset flag. These cards need not be inputted in any specific order, but for ease of checkout, it is advised to place them sequentially by their COMMON location.

5. Type 4 Card

The MOD6DF program can printout a maximum of fifty variables for each simulation. Type 4 cards are used to specify which variables are to be printed. Column 2 must contain the number four to indicate a type

4 card. When the results are printed out headings are included. These headings are designated in column 9 - 20. The exact alphanumeric title punched will be printed at the top of each page, this need not be the fortran symbol used within the program. The COMMON(3415) location of the output variable is contained in column 21 - 25 and must be right-justified.

6. Type 5 Card

Type 5 cards are used to indicate which variables are to be root-mean-squared. These cards are similar to type 1 and type 2 cards. Column 2 must contain the number five. Any pertinent information about the variable to be operated on is punched in column 5 - 20. The COMMON(3415) location of the variable must be right-justified in columns 21 - 25. The last field, column 31 - 45, indicates whether the root-mean-square operation will occur along the trajectory or at the end.

7. Type 6 Card

The type 6 card has two purposes. Its first function is trivial, but required. The number six is typed in column 2 and the rest is blank. This indicates to the computer program that there are no more input cards. Its second function involves random inputs. This card is used to indicate the number of random process (noise) generator cards that are to be read before the input process is terminated. Column 2 has the same information as before. Any pertinent information is contained in column 5 - 20. Columns 21 - 25 must be right-justified and they contain the number of random generator cards to be read.

E. AUXILIARY DECK

The auxiliary deck is a collection of subroutines and functions required by the modules. These routines are general in nature since

they can be called by several modules. They are used to calculate such things as the properties of air, engine performance, various ratios, and to locate values in the many data tables. A brief discussion of the most common subroutines is presented.

1. Boost

This subroutine calculates the thrust coefficient (CT) and the fuel flow rate (FF) during the boost phase. These values are returned to module A3 and used to calculate the missile thrust in the body axes and the missile weight.

2. Ramjet

This subroutine is used during the midcourse, cruise, phase of the missile flight. It uses a simplified ramjet model to calculate the thrust coefficient and fuel flow rate. This routine is not automatically called. The user must designate that he wishes to use it by inputting a type 3 card setting FLGRJ(606) equal to one. This then sets up the proper stepping in module A3.

3. Engine

This is the primary routine during the cruise phase. It is one of many routines within the NWC air-breathing propulsion package. This entire package is utilized to calculate the engine parameters from inlet to exhaust. Again, thrust coefficient and fuel flow rate are eventually computed and returned to module A3. The user does not need to supply any special input cards to use this subroutine. If no initial value is inputted for FLGRJ(606) it is automatically set to zero, which indicates this routine is to be used.

4. Air

All the properties associated with the air are calculated in

this subroutine. This includes the computation of the speed of sound and the dynamic pressure. Since the missile does vary in altitude, the routine takes this into account as well as the latitude. Once these values are calculated, they are returned to module G3.

5. Block Data

This routine contains data tables. These tables cover parameters from aerodynamic coefficients to thermal properties. The total data package covers specific bands within the missile operating envelope. Data is stored in matrices which includes one, two, and three dimensional ones. These data tables are readily available and there are routines designed to retrieve this information quickly.

6. Serch

This subroutine, along with several functions, is used to retrieve information from BLOCK DATA. If the present operating point of the missile is not within one of the bands of information, then the tables are interpolated. The functions THREDL, STDLI, STDLIA, and TAB do the interpolation of the tables. Since the tables are of various lengths, these functions are very general.

F. SEQUENCING

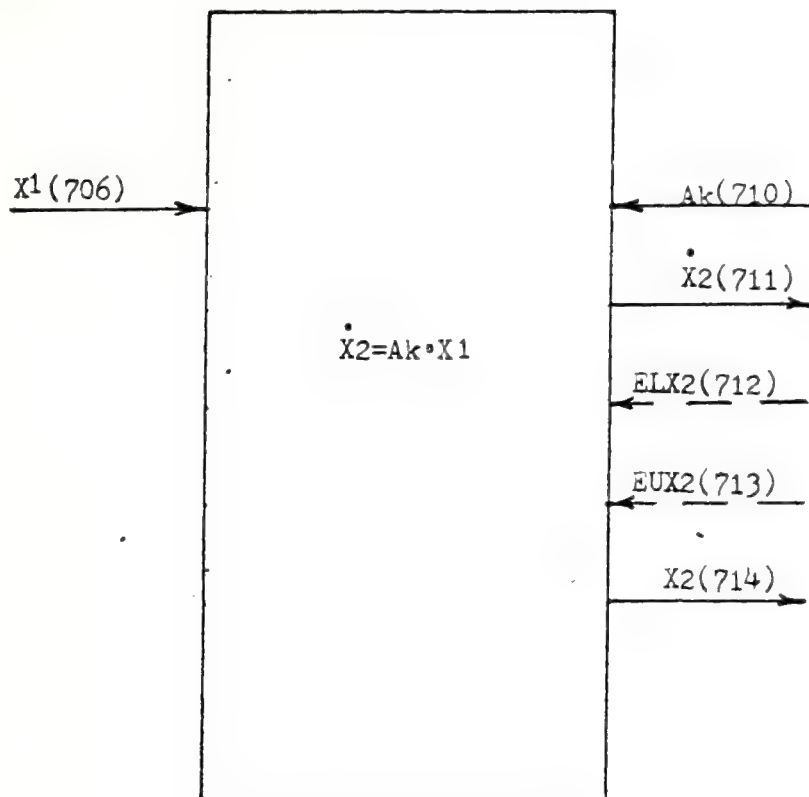
Sequencing is very important in the running of the MOD6DF program. Care must be taken to ensure that the modules are processed in the correct order at each step. This is essential to eliminate the use of obsolete values from the last cycle. An exception to this problem is the state variables. These are updated simultaneously by the integration algorithm. Any module is capable of using the most current value of these, no matter what the order of processing.

To help remedy this problem, module diagrams were devised. Module

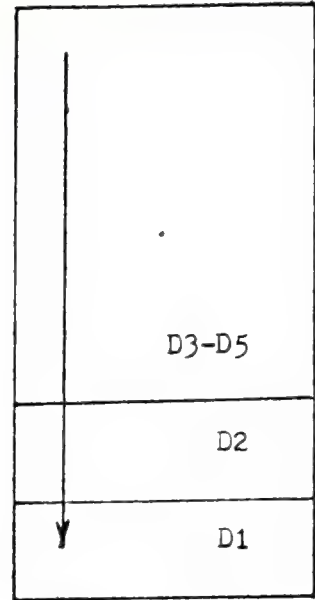
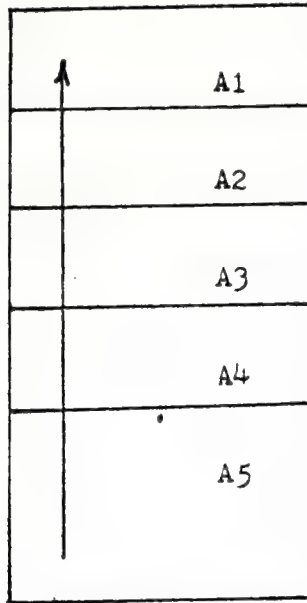
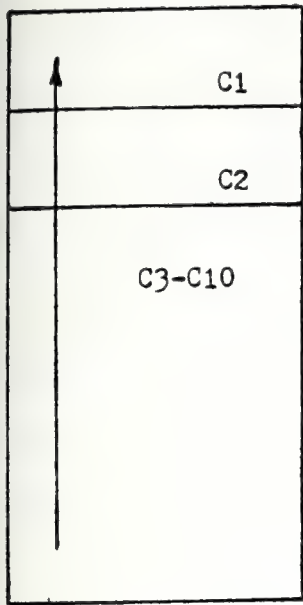
diagrams aid the user in maintaining the proper flow of variables into and out of the modules. To design a module diagram start with a box. This box will contain all the equations for a specific module. The standard procedure for showing inputs and outputs is to use arrows pointing in or out from either the left or the right side. The example in Figure 5 (Ref. 9) shows this technique. The arrows pointing in from the left indicate variable inputs from other modules. Arrows pointing out to the right indicate output going to either other modules or program output. The last set of arrows points in from the right. These show the constants brought in directly or indirectly from the initialization module. Since each arrow represents a variable, they must be defined. The usual means of labeling the arrows is to use the variable fortran symbol and in parenthesis its COMMON(3415) location.

Each variable usually has only one COMMON location associated with it. In the case of 'state' variables this is not true. State variables are defined by four consecutive COMMON locations. The first is for the derivative of the variable. The second and the third locations hold the lower and upper bounds, respectively, of the integration error. The last one contains the variable itself.

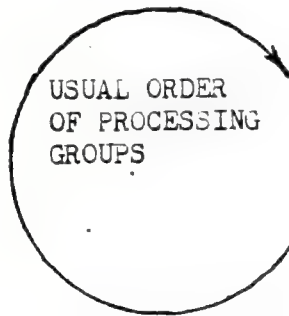
Once all the required module diagrams have been completed, they can be combined to get the overall processing order. The usual processing order, shown in Figure 6 (Ref. 10) is to start with the geophysical group, then proceed to sensors, computers, airframe, and finally dynamics. Within each group is a usual processing order and this too is shown in Figure 6. This process for determining the program sequence will eliminate the use of any obsolete values in the computer run.



MODULE DIAGRAM
FIGURE 5

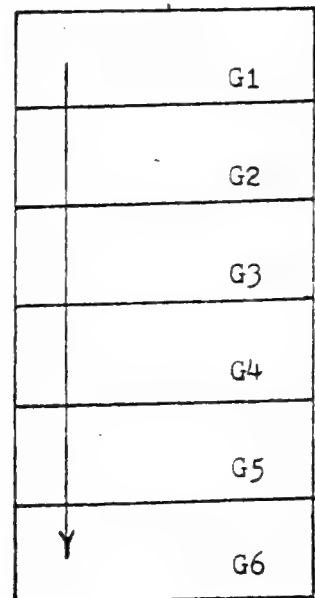
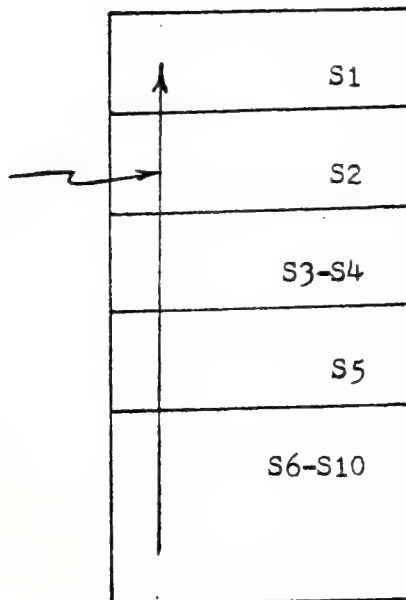


END CYCLE



BEGIN CYCLE

USUAL
ORDER OF
PROCESSING
MODULES



PROCESSING SEQUENCE

FIGURE 6

IV. COMPUTER PROGRAM CHECKOUT

The basis for the research was the MOD6DF computer program from NWC China Lake. The total package received from them consisted of a listing of the program and an uninterpreted deck of cards. The initial step was to input the cards in small groups into the computer and then to examine the source listing. This listing revealed that the original deck was punched in BCD. This fact was easy to determine since several characters were changed (Ref. 11). The library routine NEWDEK was used to translate the cards from BCD to EBCDIC.

Once the translation was completed, an attempt was made to compile the new deck. This produced an output which contained many syntax errors. These errors were divided into two major groups. One effected the use of quotation marks in FORMAT and comment statements. The other one, the more difficult, effected the DATA statements in the BLOCK DATA subroutine.

The problem with the DATA statements was due primarily to the difference in the compilers used. The compiler at NWC was much newer and allowed for the use of more sophisticated inputs. The compiler at NPS only allows a data set to start with the first element. This required the rewriting of many data groups. To complicate these revisions, a limit of nineteen continuation cards is also imposed. These restrictions demanded not only the rewriting of many data sets, but also the formation of two new ones.

With the corrections finally completed, the computer would then compile the program. The next step was to link all the subroutines together. The first attempt was unsuccessful. Inadvertently, the subroutine INTR20 had been omitted from the original deck. Using the program listing, the contents of INTR20 were typed and included in the

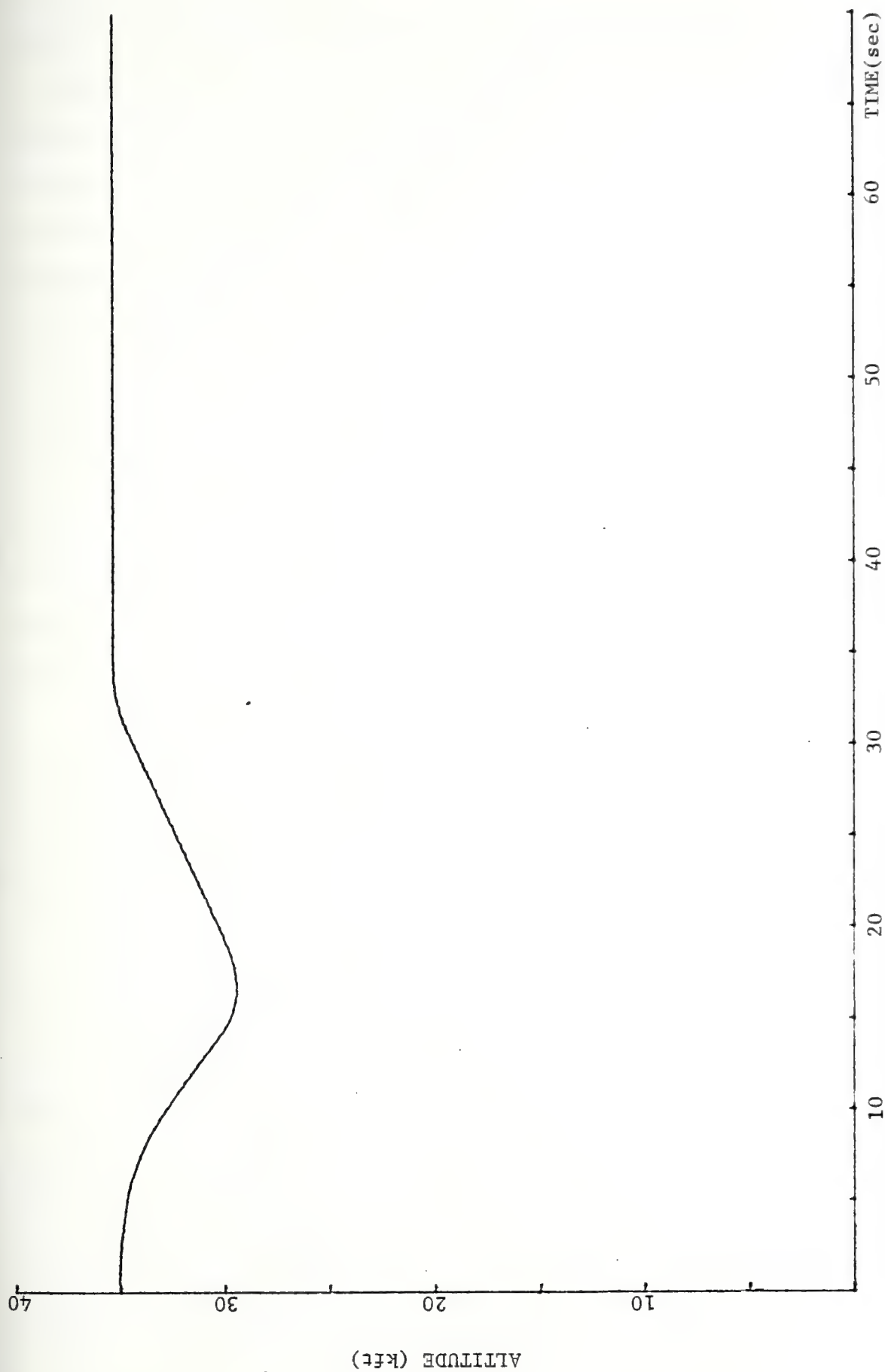
main deck. With this addition the program would now compile and link.

Now that the program would compile and link, attention was turned to getting a good simulation. To facilitate this process, two sample outputs were obtained from NWC. These outputs cover one missile flight which is broken down into a midcourse guidance and a terminal guidance simulation. Hereafter these will be referred to as midcourse baseline and terminal baseline, respectively. Using the initial conditions from the baseline models, it was hoped that the outputs could be duplicated.

A. MIDCOURSE GUIDANCE FLIGHT

The initial run, using the midcourse baseline inputs, revealed an overflow problem with the dynamic pressure (QD(508)). Using the traceback procedures outlined in the Users Manual from the W. R. Church Computer Center, the problem was confined to subroutine AIR. The problem turned out to be a translation error. The symbol $P\emptyset$ (\emptyset - zero) had translated to $P\emptyset$ in one place and P0 in another. This error caused the program to use a value left in that memory location from a previous run to calculate the dynamic pressure. With this problem remedied, the program could progress a little farther. The next stumbling block appeared as a divide check. These errors were resolved by introducing patches that would bypass a statement that tried to divide by zero. Once bypassed, that quantity would be set equal to zero. This was the normal procedure of the computer, but it would stop the run after ten such errors. Having corrected all these errors, output was obtained which covered the desired seventy seconds of flight.

When the output was examined a major switching problem within subroutine A3 (missile propulsion module) was found. The midcourse baseline utilized a simplified ramjet model, but the output was not.



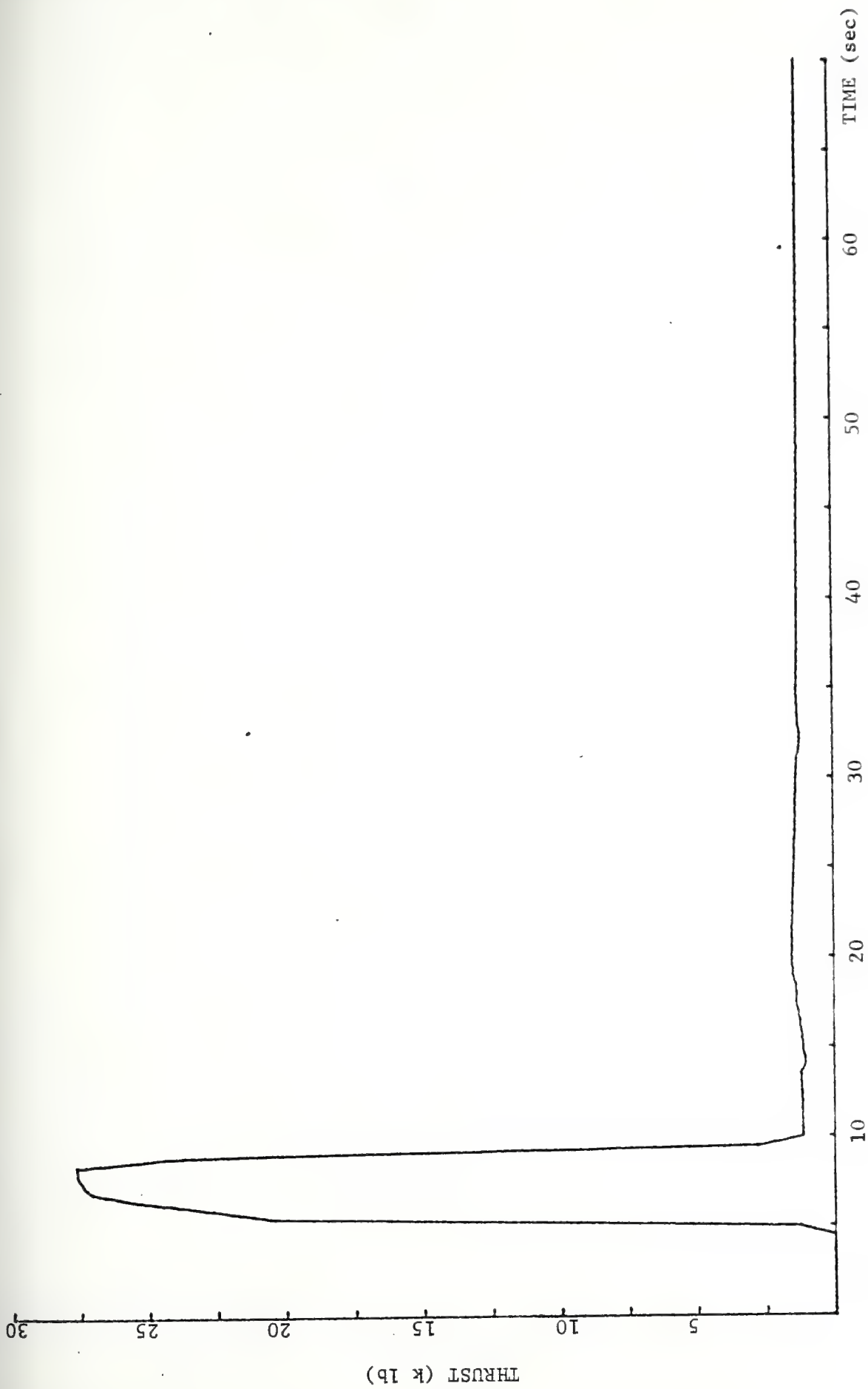
ALTITUDE VS. TIME (MIDCOURSE)

FIGURE 7

Rather than use RAMJET, the output disclosed that ENGINE had been used. The problem was in the logic statement, FLGRJ 0.0. The program was not making the desired switch to the simplified ramjet model. To remedy the problem, the data card FLGRJ(606) 1.0, was added to the input data deck. Another problem was discovered which occurred between 14.0 and 14.5 seconds. During that time period the missile angle of attack (ALPHA (330)) exceeded ten degrees. When ALPHA exceeds this angle the engine is turned off. With the engine off, no thrust is produced causing the forward velocity (VXTP(286)) to decrease. This limitation was removed from the program. After eliminating these problems a flight trajectory, Figure 7, was obtained which closely resembled the output of the mid-course baseline. A random sampling of the outputed variables were compared for exactness. The differences noted were due primarily to computer round-off error.

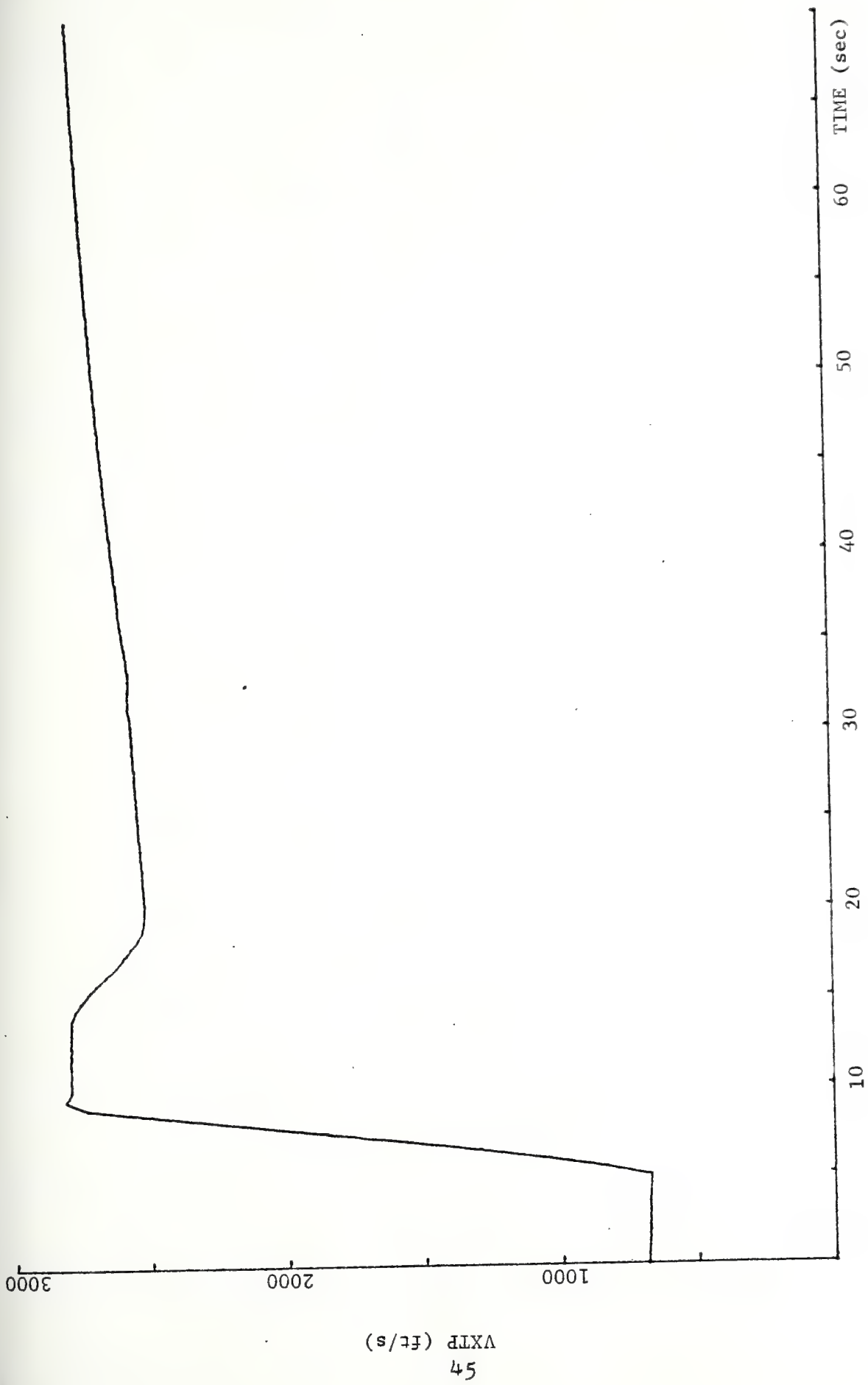
To further verify the accuracy of the two simulations, three parameter (TXBA(073), VXTP(286), and YTP(298)) were chosen as representative values for the runs. To obtain a feel for the run, these parameters are plotted in Figures 8, 9, 10. Additionally, random time samples are tabulated in Tables I, II, III.

The thrust (TXBA(073)) plot, Figure 8, can easily be divided into four mission phases. During the separation phase (0.0 - 4.5 seconds) the thrust is zero. This is expected since neither of the engines have ignited. This is followed by a rapid increase in the thrust. The maximum thrust, 27800 lbs, happens during the boost phase (4.5 - 9.5 seconds). At 9.5 seconds the boost motor stops and the transition phase occurs for the next 0.3 seconds. During this time the thrust decreases rapidly since neither engine is on. Once the engine port covers are



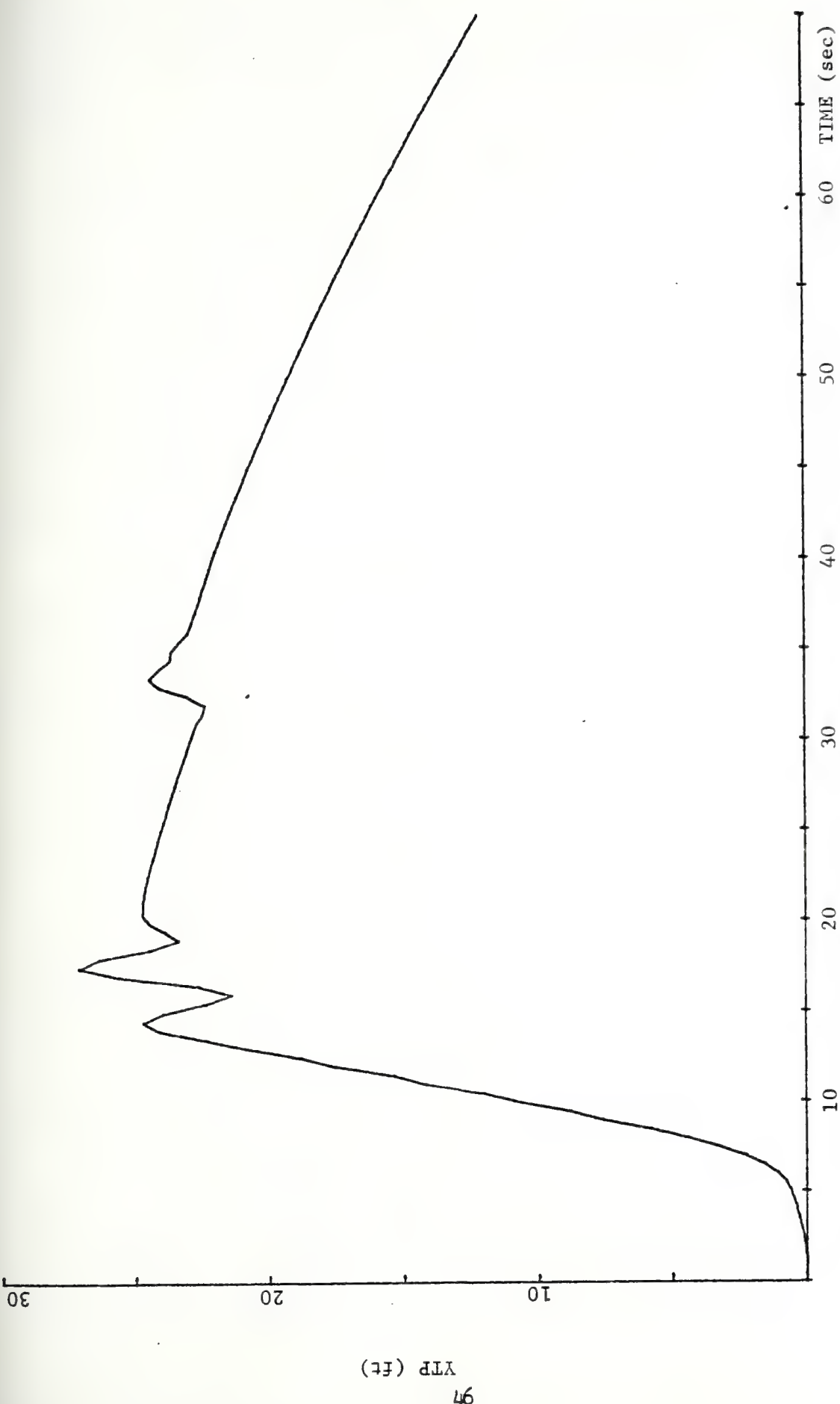
THRUST VS. TIME (MIDCOURSE)

FIGURE 8



VXTP VS. TIME (MIDCOURSE)

FIGURE 9



YTP VS. TIME (MIDCOURSE)

FIGURE 10

clear, the ramjet sustainer ignites. This action initiates the midcourse phase (9.8 - 70 seconds) and the thrust remains relatively constant at 1200 lbs.

The missile forward velocity (VXTP(286)) can be directly related to the thrust. During separation no thrust is produced, therefore the velocity is zero. As the missile is boosted to Mach two, the velocity increases and approaches its maximum value (2800 ft/s) just prior to boost engine shut-down. As the sustainer engine port covers clear, the velocity decreases slightly. Once the ramjet engine ignites the velocity profile is displayed in Figure 9 and it closely resembles that for the midcourse baseline.

The last parameter, y-displacement (YTP(298)), was included to demonstrate the accuracy of the guidance system. Figure 10 shows that prior to attaining the cruise altitude, the missile wanders off track. Once the guidance system is activated, it makes the necessary corrections to return the missile to the planned flight path. Table III shows that the y-displacement corresponds to the baseline case and at the conclusion of the run the off-track error is down to 12.03 feet.

All the information obtained from the new run was checked against the midcourse baseline. The results showed that the two simulations are within acceptable limits. With this milestone completed, the program checkout could proceed to the terminal guidance flight.

B. TERMINAL GUIDANCE FLIGHT

To checkout the terminal guidance portion of the MOD6DF program, the initial conditions from the terminal baseline were used. This required changing about a dozen input cards in the midcourse input data deck. The first run attempted turned out successful. This was due to the

TIME	0.0	7.5	10.0	20.0	25.5	30.0	40.0	50.0	60.0	70.0
BASELINE	0.0	27395	1140.6	1403.0	1377.4	1281.0	1211.3	1178.0	1148.1	1124.8
NEW	0.0	27391	1138.9	1514.3	1418.4	1314.3	1249.9	1207.0	1175.5	1149.7

TXBA (073) COMPARISON (MIDCOURSE)

TABLE I

TIME	0.0	7.5	10.0	20.0	25.0	30.0	40.0	50.0	60.0	70.0
BASELINE	681.99	1858.3	2790.2	2606.7	2604.9	2613.4	2652.4	2705.0	2743.8	2771.5
NEW	681.99	1851.7	2786.3	2506.4	2525.9	2544.5	2605.0	2665.5	2710.8	2743.9

VXTP (286) COMPARISON (MIDCOURSE)

TABLE II

TIME	0.0	7.5	10.0	20.0	25.0	30.0	40.0	50.0	60.0	70.0
BASELINE	-5E-5	3.46	11.37	27.21	25.92	24.34	20.45	16.93	13.24	9.39
NEW	-5E-5	3.19	10.68	24.43	23.93	22.82	21.96	19.14	15.81	12.03

YTP (298) COMPARISON (MIDCOURSE)

TABLE III

TIME	0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	1141.9	1131.0	1077.8	1170.9	1309.4	1501.1	1730.0	2245.3	2565.8	2691.3
NEW	1141.6	1134.8	1099.4	1212.8	1359.3	1561.7	1800.0	2305.9	2584.3	3273.6

TXBA (073) COMPARISON (TERMINAL)

TABLE IV

TIME	0.0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	2750.0	2764.2	2654.6	2471.4	2225.7	1854.5	1391.4	1376.9	1413.1	1425.05
NEW	2750.0	2761.5	2631.5	2438.7	2192.1	1828.2	1389.3	1366.7	1406.1	1465.2

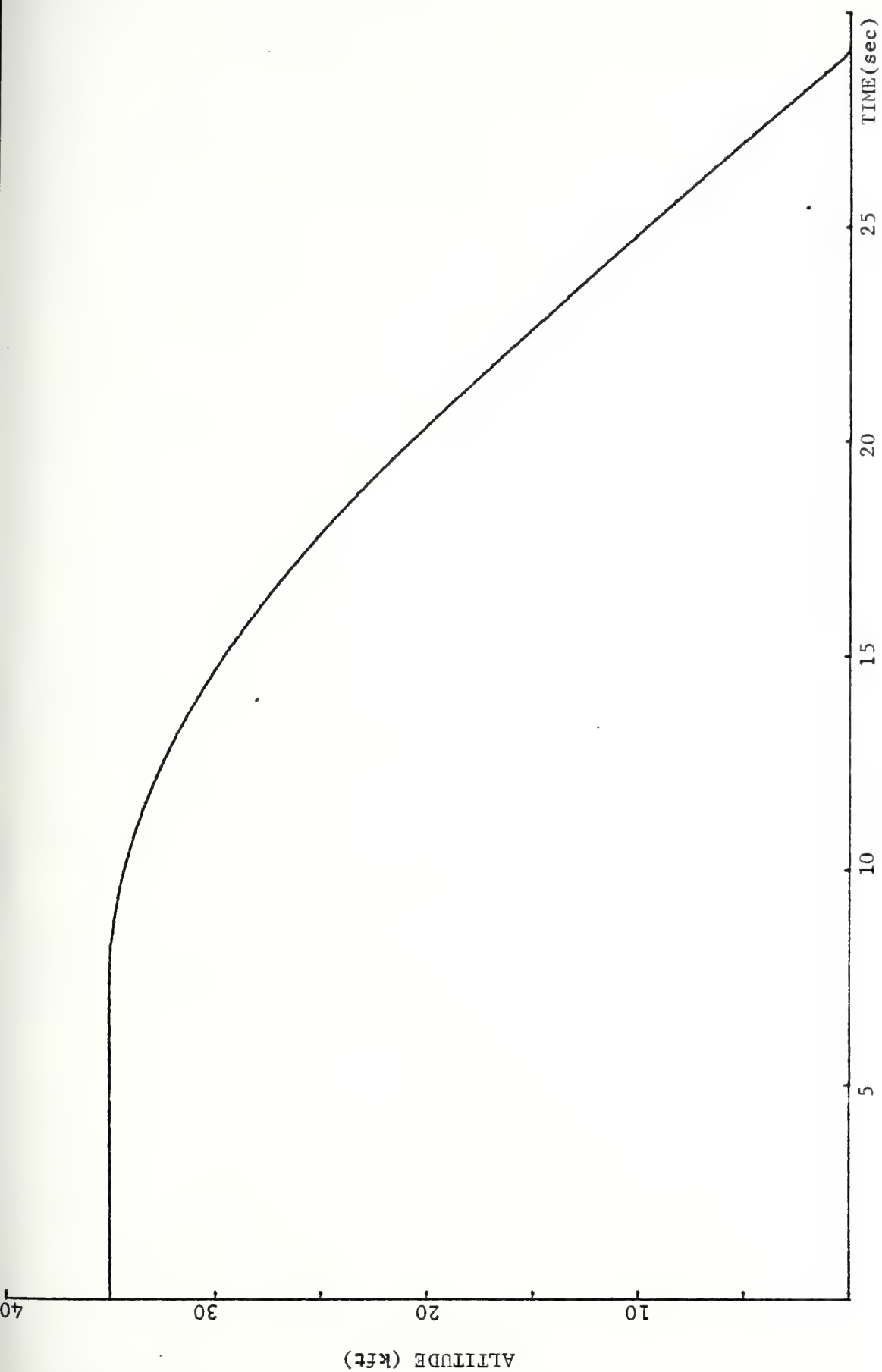
VXTP (286) COMPARISON (TERMINAL)

TABLE V

TIME	0.0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	0.0	0.0	- .4E-6	-5E-6	.7E-6	.9E-6	.1E-5	.2E-6	.1E-6	.1E-6
NEW	0.0	- .0074	.019	-1.38	-.095	-.868	-1.77	.071	-.151	-.05

YTP (298) COMPARISON (TERMINAL)

TAVLE VI



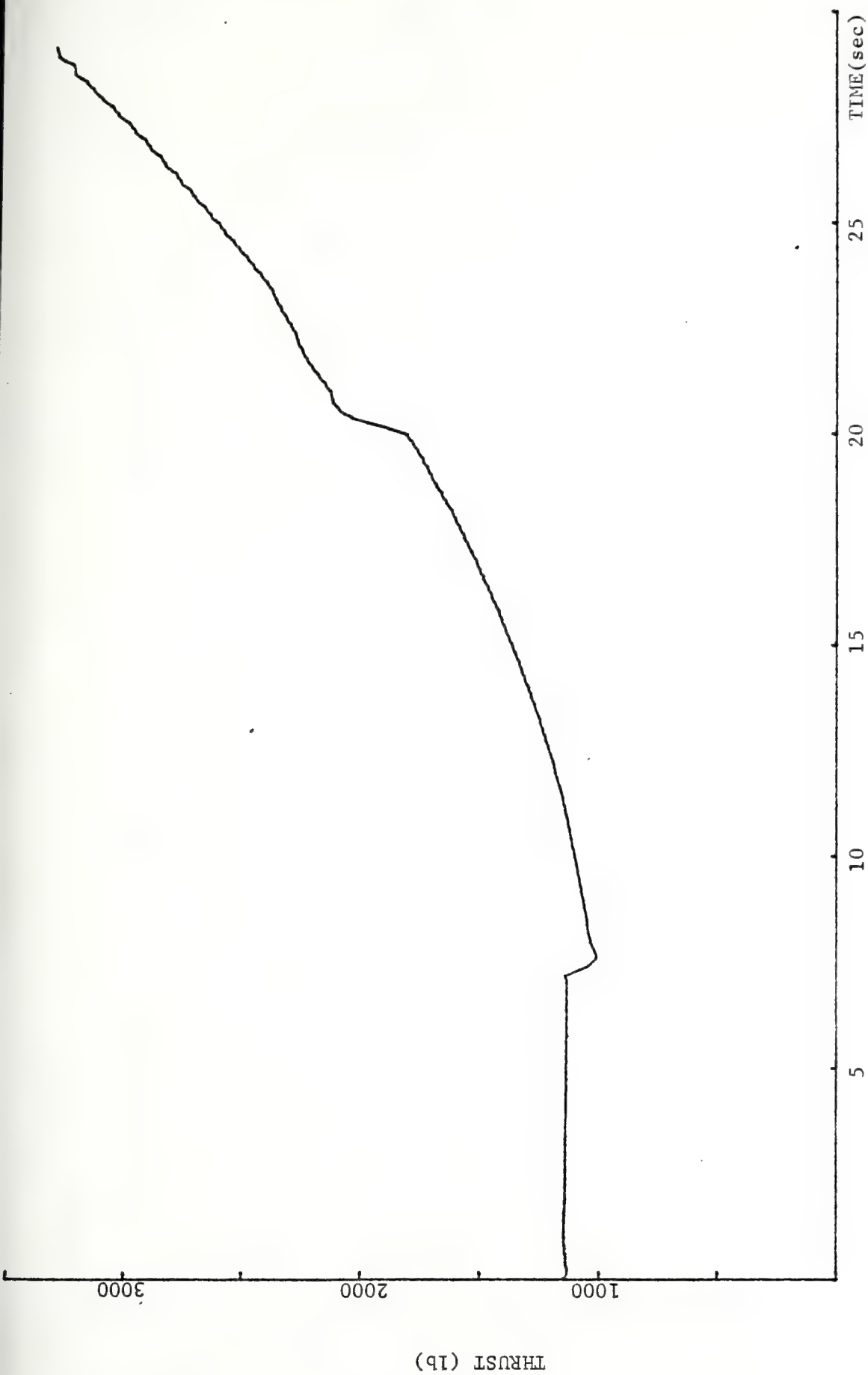
ALTITUDE VS. TIME (TERMINAL)

FIGURE 11

elimination of all the errors during the midcourse checkout. The resulting flight trajectory, Figure 11, is a very good illustration of the desired profile. To check the accuracy of the simulation, the same three parameters were chosen. Tables IV and V show that this run is almost identical to the terminal baseline. The difference is primarily due to computer round-off error. Figures 12 and 13 show the overall flight performance of the thrust and forward velocity, respectively.

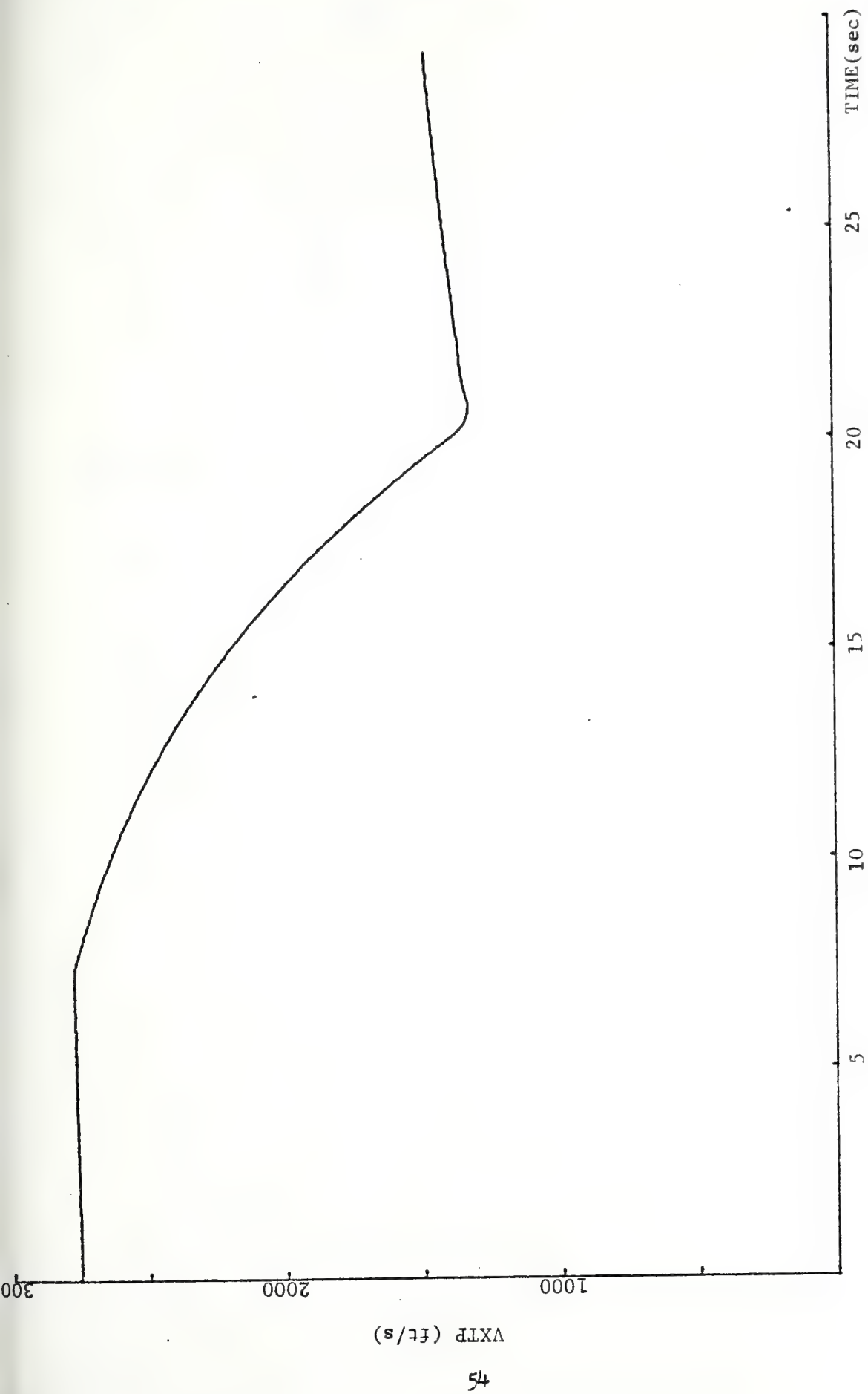
For this simulation, the y-displacement (YTP(298)) does not compare favorably with the terminal baseline. Figure 14 shows that between ten and twenty seconds in the flight, YTP develops rapid oscillations. By examining Table VI, it is noted that during this interval the two simulations differ the most. No reason for this discrepancy could be found. These oscillations do effect the flight by increasing the time of flight from 25.825 to 29.09 seconds and by increasing the miss distance. The imposed time constraints force this problem to be overlooked and to direct attention to the accuracy of the impact. Since the impact is within 0.5 feet of the target location, it is concluded that the terminal guidance simulation works properly.

With both segments of the MOD6DF computer program working correctly, thoughts could now turn towards the actual experimental runs. The results of these runs are discussed in the next section.



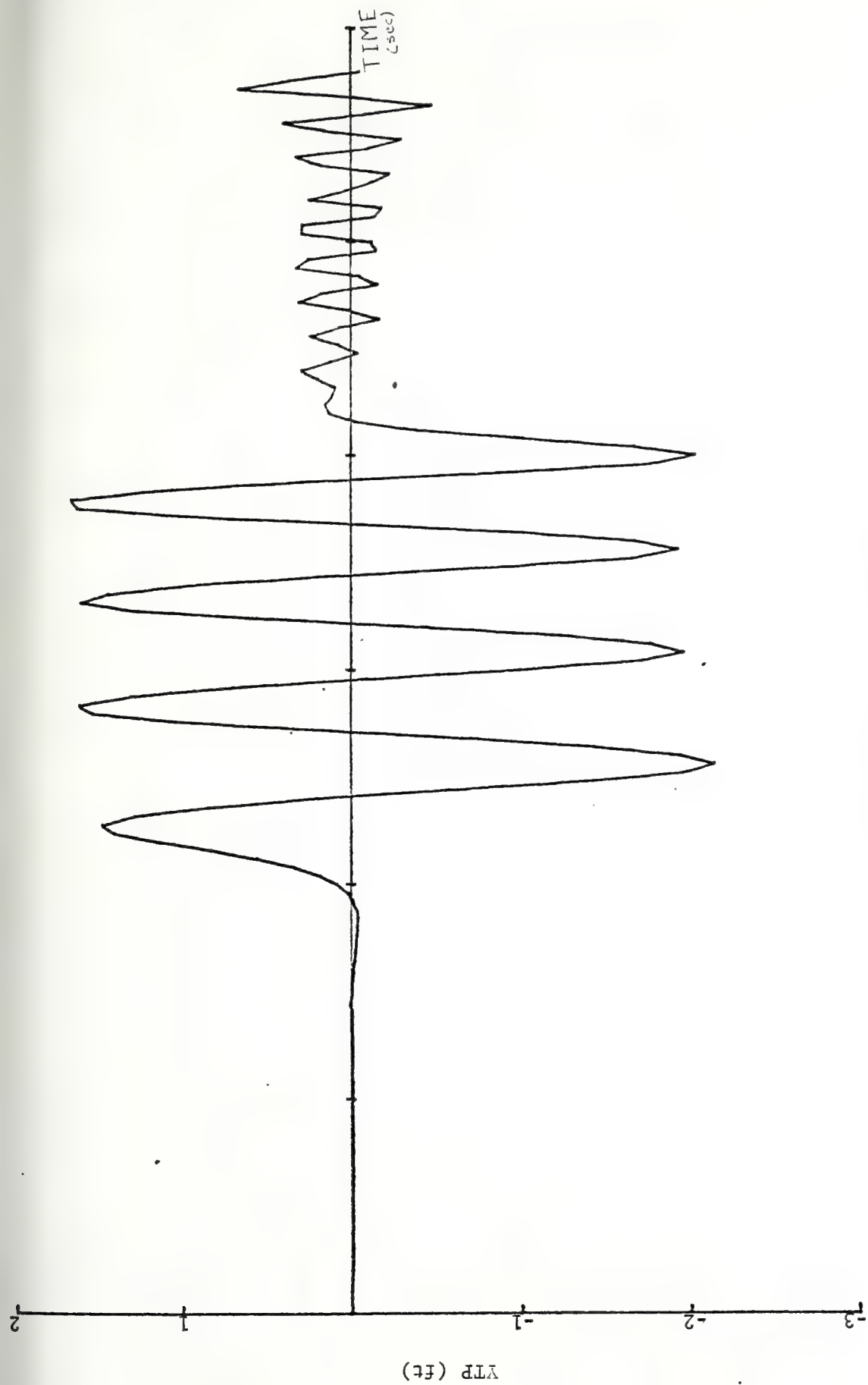
THRUST VS. TIME (TERMINAL)

FIGURE 12



VXTP VS. TIME (TERMINAL)

FIGURE 13



YTP VS. TIME (TERMINAL)

FIGURE 14

V. TERMINAL FLIGHT DISTURBANCES

With the checkout of the MOD6DF computer program completed, many ideas were discussed concerning alterations to the simulations. The three modifications decided upon were:

- * examine target miss distance when changes were made to the initial x-y-z conditions,
- * examine the terminal flight profile, range, and miss distance when the cruise altitude was reduced to approximate a sea skimming mode,
- * examine target miss distance when random noise is applied to the missile homing seeker.

To understand the changes and results, one must be familiar with the frame of reference. The origin of the reference frame travels from the launch platform to the target location at the cruise altitude. This is called the tangent plane reference system. Displacements in the x-direction (XTP(290)) are measured from the launch platform in the direction of the target. Y-displacements (YTP(298)) are measured left or right of the ideal flight path, in the tangent plane. Any vertical displacement (ZTP(306)) is measured normal to the ideal flight path, with down being the positive direction. Using this frame of reference the information in Table VII is easier to understand.

The first modifications demonstrate the effect of changing YTP. YTP was increased until a result was reached that was unsatisfactory. From the results in Table VIII, the maximum value of YTP was determined to be 1800 feet. This conclusion corresponds with the results in reference 13. Since the missile is symmetric, it was also concluded that moving YTP either right or left would give the same results.

The fifth run simulated a drop in the missile's altitude. ZTP was

RUN	XTP	YTP	ZTP
BASELINE	173552.5	0.0	0.0
1	173552.5	1000.0	0.0
2	173552.5	1500.0	0.0
3	173552.5	1800.0	0.0
4	173552.5	2000.0	0.0
5	173552.5	0.0	2000.0

INITIAL CONDITIONS

TABLE VII

RUN	XTP	YTP	ZTP	IMPACT TIME
BASELINE	234999.56	-.0506363	34999.34	28.825
1	234999.81	.1259258	34999.836	29.108
2	235000.19	.11479032	35000.32	29.124
3	235000.25	.15263242	35000.391	29.134
4	269802.0	2000.0	19705.00	35.0
5	234999.63	.45691	35000.426	30.365

IMPACT RESULTS

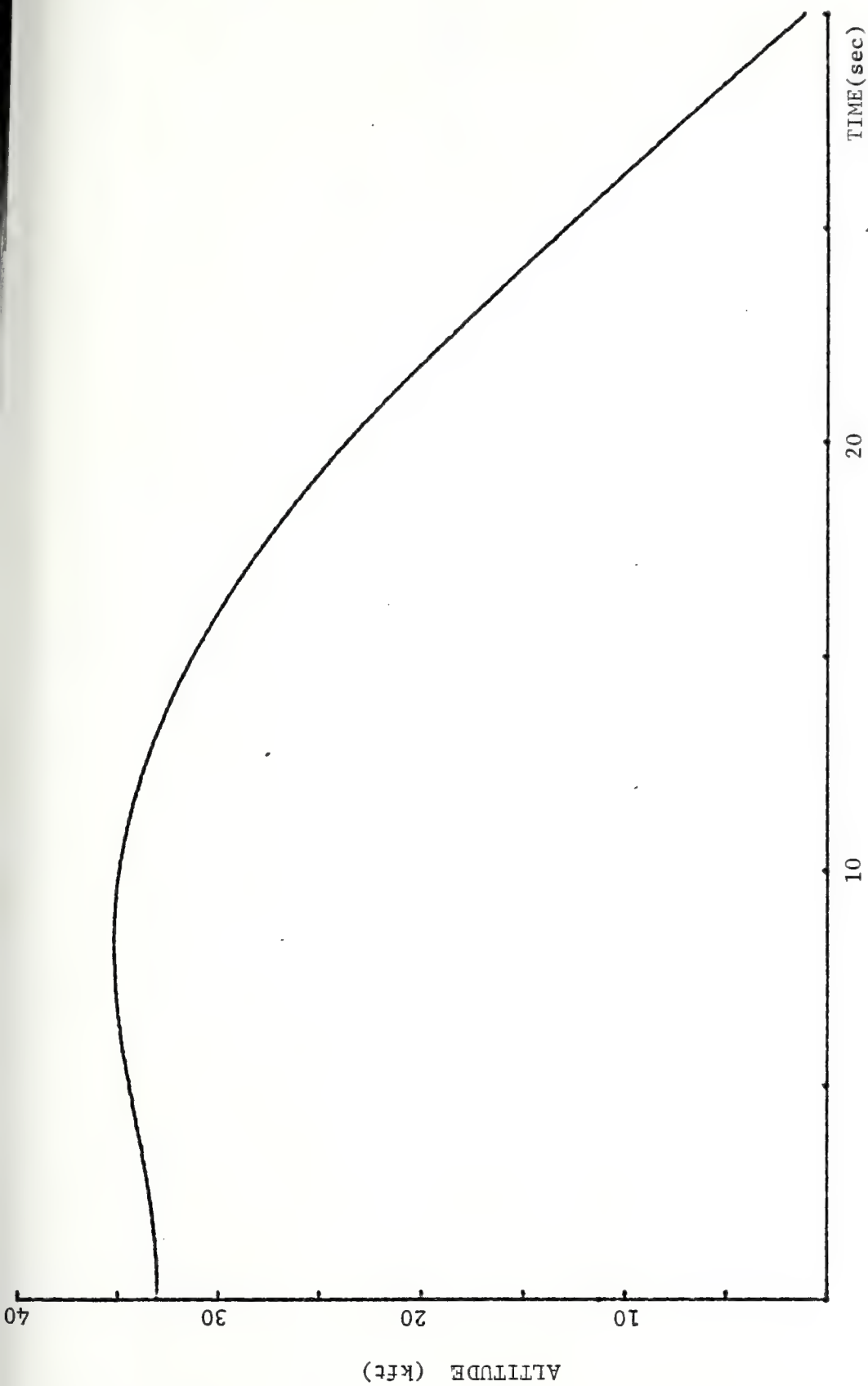
TABLE VIII

inputed as 2000.0 feet. As the simulation progressed, the missile developed the necessary commands to climb and regain the desired cruise altitude (Figure 15). It then commenced its terminal dive. The target was detected, acquired, and impact followed. This initial condition modification produced the largest miss distance (0.457 ft.).

Figure 16 was presented to show the relative locations of the miss distances for each change. Table VIII reveals that up until 1800 feet, changes to YTP created very little variation in the miss distance. It should be noted that the larger the displacement became, the greater the time until impact. The increase in time was necessary to allow the missile to acquire the target and then compute the required actuator commands to impact the target. This concluded the investigation of the effects of initial displacement error on target miss distance.

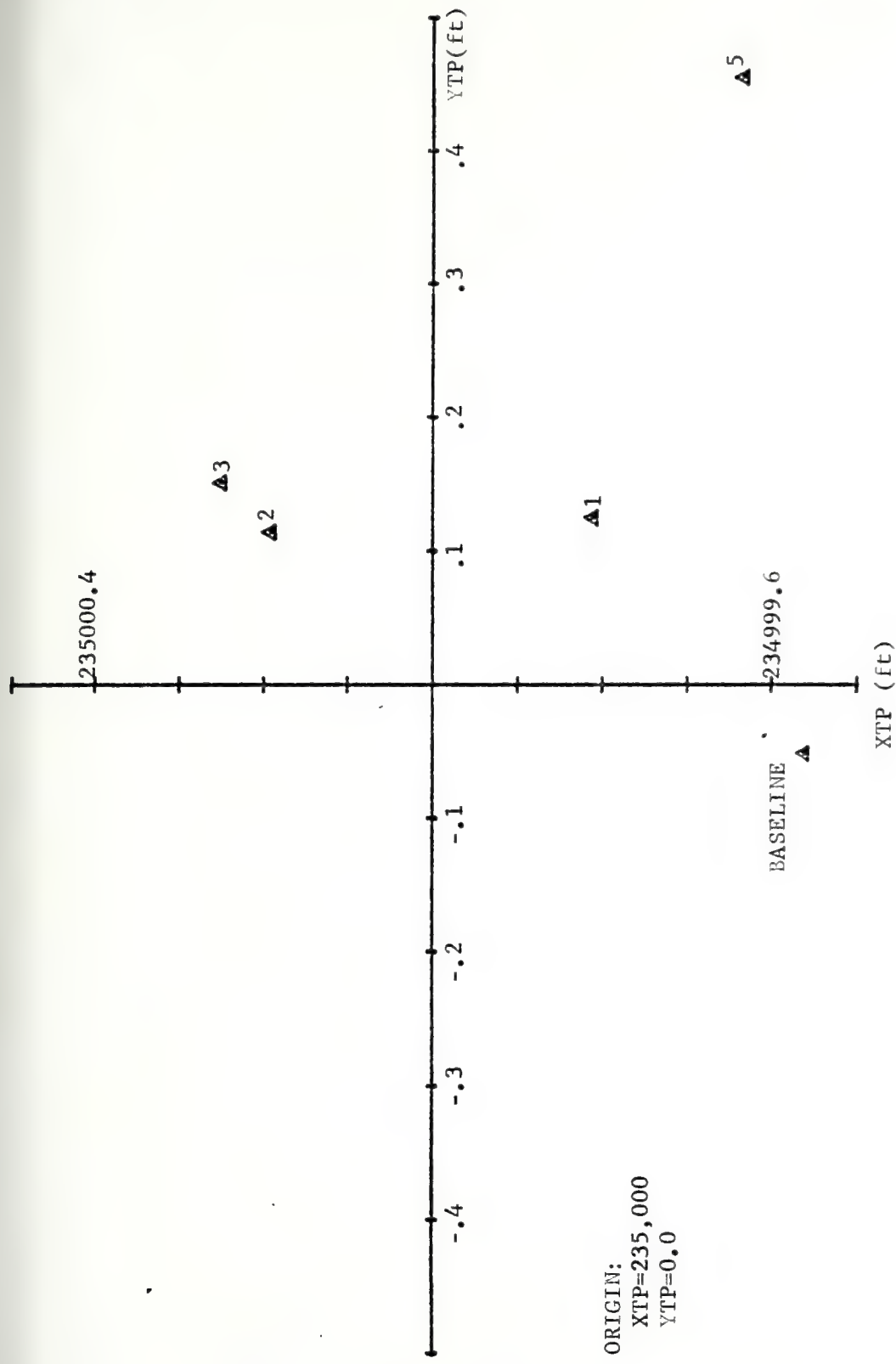
The idea behind changing the cruise altitude was to try and simulate a sea skimming profile. RAMJET contains tables that allow for four cruise altitudes. Of these four altitudes, the lowest (500 ft) was chosen even though it is high for a sea-skimmer. To run this simulation only two input data card changes were required. HO(414) and HREF(501) had to be set equal to the desired altitude. The resultant flight path is shown in Figure 17.

The simulation produced an error-free output, but at first glance the results appeared unacceptable. Still trying to compare miss distances, the downrange distance (XTP) was found to be 197916 feet. Comparing this to the terminal baseline (234999) resulted in an extreme error. It was then realized that the missile must expend more fuel at this altitude to attain the same speed. Therefore, the results could be correct. As further verification of the correctness of the run, it was assumed that the missile weight at impact should be fairly equal. The



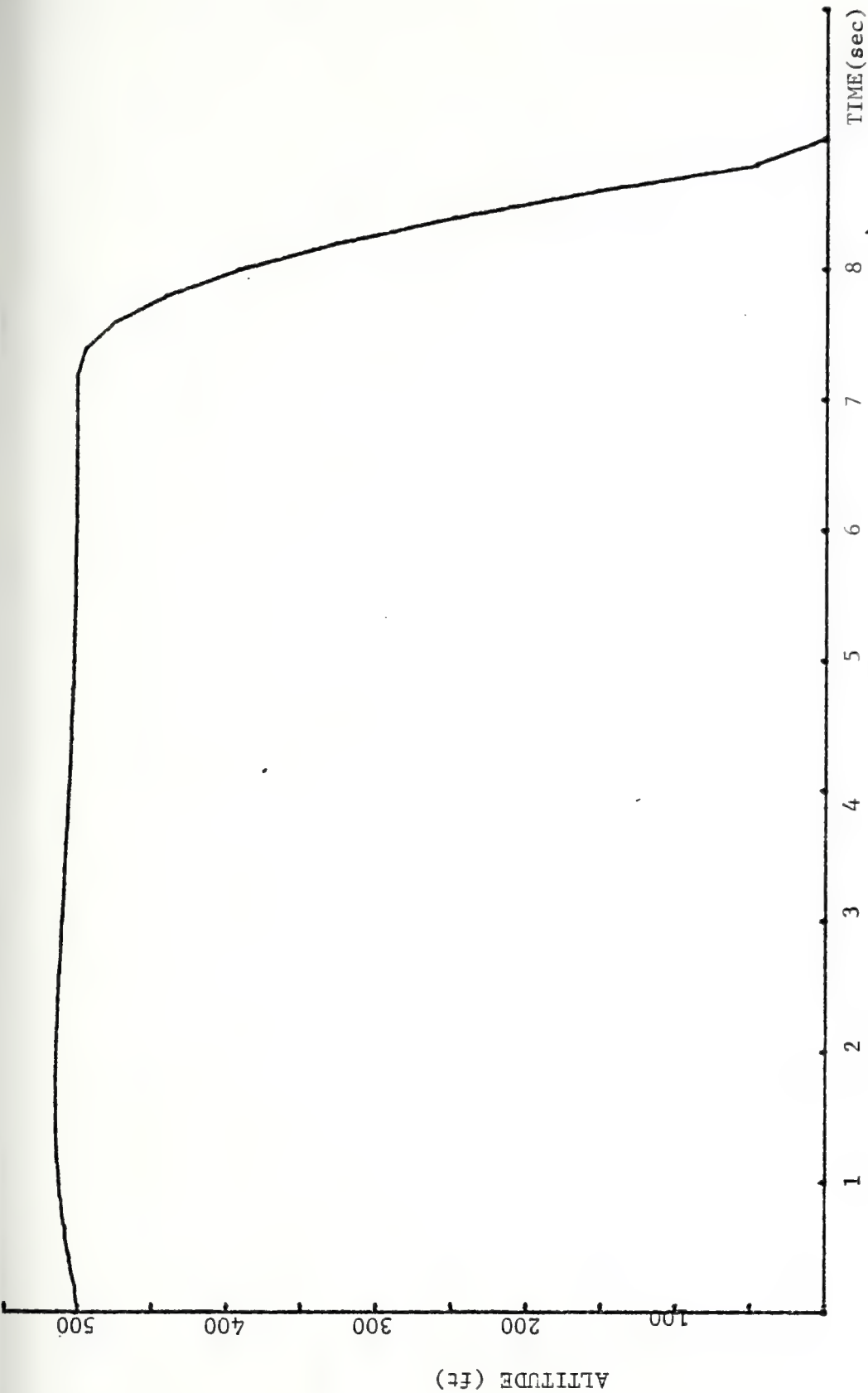
ALTITUDE VS. TIME (TERMINAL - ZTP)

FIGURE 15



IMPACT RESULTS

FIGURE 16



ALTITUDE VS. TIME (SEA-SKINNER)

FIGURE 17

simulation impact weight was 1126.34 lbs while the terminal baseline weighed 1113.12 lbs. Since no conflicting information could be found in reference 12, it was concluded that this simulation was correct.

The final flight disturbance investigated was the addition of random noise to the missile homing seeker. This simulation proved to be unsuccessful due to invalid input format. The information describing the process is brief and difficult to understand. Further information from NWC China Lake is required to be able to successfully run simulations with random noise.

With the completion of these modifications, many unanswered problems and questions still exist. The two most critical problems concern the input of random noise generators and the missile exceeding the angle of attack limitation when using ENGINE for cruise propulsion. However, the simulations did show that the MOD6DF program runs correctly at alternate cruise altitudes and with initial displacement errors.

VI. CONCLUSION

The MOD6DF computer program from NWC China Lake was converted to operate on the IBM-360 computer at the Naval Postgraduate School. The program would only function properly when using the simplified ramjet model. When this model was not used, the missile angle of attack exceeded the maximum limit of ten degrees. This error caused the ramjet engine to flame out.

Target impact errors for the terminal guidance problem were investigated when the initial displacement was modified. These modifications demonstrate the missile's accuracy when removed from the ideal flight path. The results also point out that if the target falls within the seeker search pattern, target impact will inevitably happen.

Random noise generation is possible with the MOD6DF program. However, more information discussing the parameters required to develop the noise is necessary. Additionally, sample noise inputs should be obtained that reflect the alteration to the noise subroutines.

This research involved the preliminary investigation of the MOD6DF program. The program has many possible areas, concerning guidance and control of tactical missiles, which could be developed for future study. These areas not only include the unresolved problems encountered during this research. Additionally, reference 14 and 15 contain many examples of possible advanced guidance concepts.

COMPUTER OUTPUT

[illegible]

[illegible]

COMPUTER PROGRAM

THIS IS THE MAIN MCD6DF PROGRAM

```

COMMON C(3415), TFMPS(1500)
EQUIVALENCE(C(3315),N),HMIN)
EQUIVALENCE(C(2911),HMAX)
EQUIVALENCE(C(2912),FMX)
EQUIVALENCE(C(2913),FMX(1))
EQUIVALENCE(C(3014),VAR(1))
EQUIVALENCE(C(3115),EL(1))
EQUIVALENCE(C(3216),EL(1))
EQUIVALENCE(C(3316),IPL(1))
EQUIVALENCE(C(932),T)
EQUIVALENCE(C(2904),KSTEP), (C(2905),STEP)
EQUIVALENCE(C(2907),LSTEP)
EQUIVALENCE(IPL(100))
DIMENSION DER(101)
DIMENSION VAR(101)
DIMENSION EL(100)
DIMENSION N=M
CALL ZEROT
CALL CINPT1
LSTEP = SLE11
CALL AUX1
CALL SUBL2,N
DO 60 I=1,N
  J = IPL(I-1)
  EL(I-1) = C(J+1)
  EL(I) = C(J+2)
  VAR(I) = C(J+3)
  DER(I) = C(J)
  VAR(I) = T
  CALL AUXSUB
  CALL ANFK
  DO 50 I=2,N
    J = IPL(I-1)
    C(J+3) = VAR(I)
    T = VAR(I)
  CALL SUBL3
  IF (KSTEP.EQ. 1) GC TC 1007
  IF (KSTEP - 1) 70,1007,7C
  CALL PRECES
  CALL RESET
  GO TC (1000,1001,1002,1003,1004,1005,1006,1007,1008,1009,1010),

```

CC CC

```

100C CALL ZEROT
1001 CINPT1
1002 LSTEP = SLE11
1003 CALL AUX1
1004 CALL SUBL2,N
1005 DO 60 I=1,N
      J = IPL(I-1)
      EL(I-1) = C(J+1)
      EL(I) = C(J+2)
      VAR(I) = C(J+3)
      DER(I) = C(J)
      VAR(I) = T
      CALL AUXSUB
      CALL ANFK
      DO 50 I=2,N
        J = IPL(I-1)
        C(J+3) = VAR(I)
        T = VAR(I)
      CALL SUBL3
      IF (KSTEP.EQ. 1) GC TC 1007
      IF (KSTEP - 1) 70,1007,7C
      CALL PRECES
      CALL RESET
      GO TC (1000,1001,1002,1003,1004,1005,1006,1007,1008,1009,1010),

```


MCD60530
 MCD60540
 MCD60545
 MCD60550

1 LSTEP
 1010 CALL EXIT
 STCP
 END

SUBROUTINE ZERO

COMMON C(2415)
 DO 1 KLEAF = 1, 4915
 1 C(KLEAF) = 0.0
 RETURN
 END

SUBROUTINE QINPT1

BASIC INFLT SUBROUTINE CINPT1
 SUBROUTINE QINPT1 IS THE BASIC INPUT SUBROUTINE

DIMENSION LISTNO(50), VALUE(50)
 DIMENSION SUBNO(99), IR(2), VR(2)
 DIMENSION RNDMNO(50)
 DIMENSION ALPHA(4), CNAME1(50), CNAME2(50), CNAME3(50), OUTNC(50)
 DIMENSION MCCNO(99)
 DIMENSION STATNO(100)

COMMON C(2415)
 REAL MCDNC
 INTEGER CLTNC
 INTEGER RNDMNO
 INTEGER STATNG
 EQUIVALENCE (C(2442), LOSTAT)
 EQUIVALENCE (C(2200), STATNO(1)), (C(2441), NOSTAT)
 EQUIVALENCE (C(2801), NOSUB), (C(2802), SUBNO(1)), (C(2663), IR(1)),
 1 (C(2665), VR(1))
 EQUIVALENCE (C(2701), NCMOD)
 EQUIVALENCE (C(2661), NCOU)
 EQUIVALENCE (C(2448), NORNDM)
 EQUIVALENCE (C(2300), NOLIST), (C(2301), LISTNO(1)), (C(2351),
 1 VALUE(1))
 EQUIVALENCE (C(2500), CUTNO(1))
 EQUIVALENCE (C(2550), CNAME1(1))
 EQUIVALENCE (C(2600), CNAME2(1))
 EQUIVALENCE (C(2150), CNAME3(1))
 EQUIVALENCE (C(2450), RNDMNO(1))
 EQUIVALENCE (C(2702), MCDNO(1))
 JAR = C

CINP0030
 CINP0010
 CINP0020
 CINP0230
 CINP0240
 CINP0250
 CINP0300
 CINP0040
 CINP0310
 CINP0320
 CINP0330
 CINP0340
 CINP0050
 CINP0110
 CINP0120
 CINP0130

CINP0350


```

1 READ(5,2) IR(1),ALPHA(1),ALPHA(2),ALPHA(3),ALPHA(4),IR(2),VR(1),
1 VR(2)
1 WRITE(6,2) IR(1),ALPHA(1),ALPHA(2),ALPHA(3),ALPHA(4),IR(2),VR(1),
1 VR(2)
2 FORMAT(I2,2X,4A4,I5,5X,2E15.8)
IF (IR(1) .NE. 0) GO TO 7
IF (IR(1)) 7,18,7
18 IF = IR(2)
REWIND
I2 = VR(1) - 1.
I3 = VR(2)
IF (I2) 11,15,14
DO 8 I = 1,I2
14 READ (I1,10) J,X
15 DO 9 I = 1,I3
15 READ (I1,2) J,X
9 C(J) = X
10 FORMAT (I5,E15.9)
GO TO 1
11 JAR = 1
GO TO 1
7 IF (IR(1) .NE. 1) GO TO 3
7 IF (IR(1) - 1) 3,19,3
19 NOSUB = NCSUB + 1
SUBNC(NCSUB) = IR(2)
GO TO 1
3 IF (IR(1) .NE. 2) GO TO 4
3 IF (IR(1) - 2) 4,20,4
20 NOMOD = NCMOD + 1
MODNC(NCMOD) = IR(2)
GO TO 1
4 IF (IR(1) .NE. 3) GO TO 5
4 IF (IR(1) - 3) 5,21,5
21 L = IR(2)
C(L) = VR(1)
IF (JAR .EQ. 1) WRITE (I1,10)L,VR(1)
IF (JAR - 1) 23,22,23
22 WRITE (I1,10)L,VR(1)
IF (VR(2) .EQ. 0) GO TO 1
23 IF (VR(2)) 24,1,24
24 NOLIST = NOLIST + 1
LISTNC(NCLIST) = L
VALUE(NCLIST) = VR(1)
GO TO 1
5 IF (IR(1) .NE. 4) GO TO 6
5 IF (IR(1) - 4) 6,25,6
25 NOCUT = NCUT + 1
UNAME1(NCUT) = ALPHA(2)

```

CINP0410
CINP0420
CINP0430
CINP0440
CINP0450
CINP0460
CINP0470
CINP0500
CINP0510
CINP0520
CINP0530
CINP0540
CINP0550
CINP0560
CINP0570
CINP0600
CINP0610
CINP0620
CINP0630
CINP0640
CINP0650
CINP0660
CINP0670
CINP0700
CINP0710
CINP0720
CINP0730
CINP0740
CINP0750
CINP0760
CINP0770
CINP1000
CINP1010
CINP1020
CINP1030
CINP1040
CINP1050
CINP1060
CINP1070
CINP1100
CINP1120

SUBL00200
SUBL00210
SUBL00220
SUBL00230
SUBL00240
SUBL00250
SUBL00260
SUBL00270
SUBL00280
SUBL00290
SUBL00300
SUBL00310
SUBL00320

SUBL0010
SUBL00050
SUBL00020
SUBL00030

SUBL00060
SUBL00070
SUBL00100
SUBL00110
SUBL00120
SUBL00130
SUBL00140
SUBL00150
SUBL00160
SUBL00170
SUBL00200
SUBL00210
SUBL00220
SUBL00230
SUBL00240
SUBL00250
SUBL00260
SUBL00270
SUBL00300
SUBL00310
SUBL00320

SUBL0010
SUBL00050
SUBL00020

6 GO TC 1 RCM1
7 GO TC 1 ALXA1
8 GO TC 1 ALXB1
9 GO TC 1 ALXC1
1 CONTINUE
RETURN
END

C
C
C
C

SUBROUTINE SUBL2
DIMENSION SUBNO(99)
COMMON C(3415) (C(2801),NCSUB)
EQUIVALENCE (C(2802),SUBNO(1))
DO 1 I = 1, NCSUB
J = SUBNO(I)
GO TO (1, 2, 3, 4, 5, 6, 7, 8, 9), J
2 GO TC 1 INFT2
3 GO TC 1 CCFT2
4 GO TC 1 STGE2
5 GO TC 1 CNTR2
6 GO TC 1 RCM2
7 GO TC 1 ALXA2
8 GO TC 1 ALXB2
9 GO TC 1 ALXC2
1 CONTINUE
RETURN
END

C
C
C
C

SUBROUTINE SUBL3
DIMENSION SUBNO(99)
COMMON C(3415)


```

EQUIVALENCE (C(2801),NOSUB)
EQUIVALENCE (C(2802),SUENO(1))
DO 1 I = 1, NOSUB
  J = SUENO(I)
  GO TC (1, 2, 3, 4, 5, 6, 7, 8, 9), J
2 CALL INPT3
3 GO TC 1
4 CALL CUF3
5 GO TC 1
6 CALL STGE3
7 GO TC 1
8 CALL CNTR3
9 GO TC 1
10 CALL RNDM3
11 GO TC 1
12 CALL ALXA3
13 GO TC 1
14 CALL ALXB3
15 GO TC 1
16 CALL ALXC3
17 CONTINUE
18 RETURN
19 END

```

CCC C

```

SUBROUTINE AUXI
  DIMENSION MCDNO(99)
  COMMON C(3415)
  REAL MCENOE (C(2701), NCMOD)
  EQUIVALENCE (C(2702), MCDNO(1))
  EQUIVALENCE (C(2659), CNCE)
  EQUIVALENCE (C(3315), N)
  N = 1
  CNCE = 0
  DO 1 I = 1, NCMOD
    L = MCENOE(I)
    GO TC (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37), L
1 CALL AI
2 GO TC 1
3 CALL A2I
4 GO TC 1
5 CALL A3I
6 GO TC 1
7 CALL A4I
8 GO TC 1
9 CALL A5I
10 GO TC 1
11 CALL A6I
12 GO TC 1
13 CALL A7I
14 GO TC 1
15 CALL A8I
16 GO TC 1
17 CALL A9I
18 GO TC 1
19 CALL A10I
20 GO TC 1
21 CALL A11I
22 GO TC 1
23 CALL A12I
24 GO TC 1
25 CALL A13I
26 GO TC 1
27 CALL A14I
28 GO TC 1
29 CALL A15I
30 GO TC 1
31 CALL A16I
32 GO TC 1
33 CALL A17I
34 GO TC 1
35 CALL A18I
36 GO TC 1
37 CALL A19I
38 GO TC 1
39 CALL A20I
40 GO TC 1
41 CALL A21I
42 GO TC 1
43 CALL A22I
44 GO TC 1
45 CALL A23I
46 GO TC 1
47 CALL A24I
48 GO TC 1
49 CALL A25I
50 GO TC 1
51 CALL A26I
52 GO TC 1
53 CALL A27I
54 GO TC 1
55 CALL A28I
56 GO TC 1
57 CALL A29I
58 GO TC 1
59 CALL A30I
60 GO TC 1
61 CALL A31I
62 GO TC 1
63 CALL A32I
64 GO TC 1
65 CALL A33I
66 GO TC 1
67 CALL A34I
68 GO TC 1
69 CALL A35I
70 GO TC 1
71 CALL A36I
72 GO TC 1
73 CALL A37I
74 GO TC 1
75 CALL A38I
76 GO TC 1
77 CALL A39I
78 GO TC 1
79 CALL A40I
80 GO TC 1
81 CALL A41I
82 GO TC 1
83 CALL A42I
84 GO TC 1
85 CALL A43I
86 GO TC 1
87 CALL A44I
88 GO TC 1
89 CALL A45I
90 GO TC 1
91 CALL A46I
92 GO TC 1
93 CALL A47I
94 GO TC 1
95 CALL A48I
96 GO TC 1
97 CALL A49I
98 GO TC 1
99 CALL A50I

```


5	CALL A4I
6	GO TC A5I
7	CALL C1I
8	GO TC C2I
9	CALL C3I
10	GO TC C4I
11	CALL C5I
12	GO TC C6I
13	CALL C7I
14	GO TC C8I
15	CALL C9I
16	CALL C1CI
17	GO TC D1I
18	CALL C2I
19	GO TC C3I
20	CALL C4I
21	GO TC D5I
22	CALL G1I
23	GO TC G2I
24	CALL G3I
25	GO TC G4I
26	CALL G5I
27	GO TC G6I
28	CALL S1I

ALXI	023C
ALXI	024C
ALXI	025C
ALXI	026C
ALXI	027C
ALXI	030C
ALXI	031C
ALXI	032C
ALXI	033C
ALXI	034C
ALXI	035C
ALXI	036C
ALXI	037C
ALXI	040C
ALXI	041C
ALXI	042C
ALXI	043C
ALXI	044C
ALXI	045C
ALXI	046C
ALXI	047C
ALXI	050C
ALXI	051C
ALXI	052C
ALXI	053C
ALXI	054C
ALXI	055C
ALXI	056C
ALXI	057C
ALXI	060C
ALXI	061C
ALXI	062C
ALXI	063C
ALXI	064C
ALXI	065C
ALXI	066C
ALXI	067C
ALXI	070C
ALXI	071C
ALXI	072C
ALXI	073C
ALXI	074C
ALXI	075C
ALXI	076C
ALXI	077C
ALXI	100C
ALXI	101C
ALXI	102C

ALXSI103C
 ALXSI104C
 ALXSI105C
 ALXSI106C
 ALXSI107C
 ALXSI110C
 ALXSI111C
 ALXSI112C
 ALXSI113C
 ALXSI114C
 ALXSI115C
 ALXSI116C
 ALXSI117C
 ALXSI118C
 ALXSI119C
 ALXSI120C
 ALXSI121C
 ALXSI122C
 ALXSI123C
 ALXSI124C
 ALXSI125C
 ALXSI126C

ALXS001C
 ALXS012C
 ALXS013C
 ALXS014C
 ALXS015C
 ALXS016C
 ALXS017C
 ALXS018C

ALXS019C
 ALXS0110C
 ALXS0111C
 ALXS0112C
 ALXS0113C
 ALXS0114C
 ALXS0115C
 ALXS0116C
 ALXS0117C
 ALXS0118C
 ALXS0119C
 ALXS0120C
 ALXS0121C
 ALXS0122C
 ALXS0123C
 ALXS0124C
 ALXS0125C
 ALXS0126C
 ALXS0127C
 ALXS0128C

29 CALL S2I
 30 GO TC S2I
 31 CALL S3I
 32 GO TC S3I
 33 CALL S4I
 34 GO TC S4I
 35 CALL S5I
 36 GO TC S5I
 37 CALL S6I
 38 GO TC S6I
 39 CALL S7I
 40 GO TC S7I
 41 CALL S8I
 42 GO TC S8I
 43 CALL S9I
 44 GO TC S9I
 45 CALL S10I
 46 GO TC S10I
 47 CCNT INLE
 48 RETURN
 49 END

SUBROUTINE AUXSUB

DIMENSION DER(101)
 DIMENSION VAR(101)
 DIMENSION IPL(100)
 DIMENSION MCDNO(99)
 COMMON CC(3415)
 REAL MCDNCE
 EQUIVALENCE (C(2701), NOMOD)
 EQUIVALENCE (C(2702), MCDNO(1))
 EQUIVALENCE (C(3316), IPL(1))
 EQUIVALENCE (C(3014), VAR(1))
 EQUIVALENCE (C(2913), DER(1))
 EQUIVALENCE (C(932), T)
 EQUIVALENCE (C(3315), N)
 DO 50 I = 1, N
 J = IPL(I-1)
 C(J+3) = VAR(I)
 T = VAR(I)
 DO 1 I = 1, NCMCD
 L = MCDNCE(I)
 GO TC (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,
 123,24,25,26,27,28,29,30,31,32,33,34,35,36,37),L
 2 CALL AI
 3 GO TC

C C C C


```

27 CALL G6
28 GO TO 1
29 CALL S1
30 GO TC S1
31 CALL S2
32 GO TC S2
33 CALL S3
34 GO TC S3
35 CALL S4
36 GO TC S4
37 CALL S5
38 GO TC S5
39 CALL S6
40 GO TC S6
41 CALL S7
42 GO TC S7
43 CALL S8
44 GO TC S8
45 GO TO 1
46 CALL S9
47 GO TC S9
48 CALL S10
49 GO TC S10
50 CONTINUE
51 DO 60 I = 2, N
52 J = IFL(I-1)
53 DER(I) = ((J)
54 RETURN
55 END

```

SUER CUTINE AMRK

DOUBLE PRECISION VERSION. SAVES
THE INDEPENDANT VARIABLE IN
FULL D.P. FUNCTION VALUES AND

```

DIMENSION V(1), D(1), EU(1), FL(1)
DOUBBLE P1, P2, P3, P4, C2, C3, C4
COMMON T(I5C0), TME, DELT
PRECISION T(3014), V(1),
(C(2913), D(1) )
(C(3215), EU(1) )
(C(3315), FL(1) )
(C(2615), ONCE )
(C(3315), NI )
(C(2912), HMAX)
(C(2911), TMIN)
(J1, NI)

```

ARRK01140C
ARRK01150C
ARRK01160C

AMRK022200
 AMRK022300
 AMRK022400
 AMRK022500
 AMRK022600
 AMRK022700
 AMRK031000
 AMRK031100
 AMRK031200
 AMRK031300
 AMRK031400
 AMRK031500
 AMRK031600
 AMRK031700
 AMRK031800
 AMRK031900
 AMRK032000
 AMRK032100
 AMRK032200
 AMRK032300
 AMRK032400
 AMRK032500
 AMRK032600
 AMRK032700
 AMRK032800
 AMRK032900
 AMRK033000
 AMRK033100
 AMRK033200
 AMRK033300
 AMRK033400
 AMRK033500
 AMRK033600
 AMRK033700
 AMRK033800
 AMRK033900
 AMRK034000
 AMRK034100
 AMRK034200
 AMRK034300
 AMRK034400
 AMRK034500
 AMRK034600
 AMRK034700
 AMRK034800
 AMRK034900
 AMRK035000
 AMRK035100
 AMRK035200
 AMRK035300
 AMRK035400
 AMRK035500
 AMRK035600
 AMRK035700
 AMRK035800
 AMRK035900
 AMRK036000
 AMRK036100
 AMRK036200
 AMRK036300
 AMRK036400
 AMRK036500
 AMRK036600
 AMRK036700
 AMRK036800
 AMRK036900
 AMRK037000
 AMRK037100
 AMRK037200
 AMRK037300
 AMRK037400
 AMRK037500
 AMRK100000
 AMRK100100
 AMRK100200

```

1  IF (CNCE) 2,1,2
   N=NI-1
   CPT = 0
   P1 = 0.2516666666666667
   P2 = 0.4583333333333333
   P3 = 0.5416666666666667
   P4 = 0.375
   C2 = 0.7916666666666667
   C3 = 0.2083333333333333
   C4 = 0.0416666666666667
   KOUNT = 0
2  NI = N
   J2 = J1+NI
   J3 = J2+NI
   J4 = J3+NI
   J5 = J4+NI
   J6 = J5+NI
   J7 = J6+NI
   J8 = J7+NI
   J9 = J8+NI
   J10 = J9+NI
   J11 = J10+NI
   J12 = J11+NI
   J13 = J12+NI
   J14 = J13+NI
   J15 = J14+NI
   IF (CPT.NE.0.) GO TO 60
   IF (OPT) 60,49,60
   IF (DELT) 60,50,200
   IF (DELT) 50,200,50
50 KOUNT=C
   STAFF FLNGE-KUTTA INTEGRATION.
   COMPUTE KC
60 CONTINUE
   DO 25 I=1,N1
     K5=J15+I
     IF(SNGL(T(K5)) .NE. V(I+1))T(K5) = V(I+1)
     IF(SNGL(T(K5)) .NE. V(I+1)) 24,25,24
     T(K5) = V(I+1)
25 CONTINUE
   IF(SNGL(TME) .NE. V(1)) TME = V(1)
26 TME=V(1)
27 KOUNT = KOUNT + 1
70 DO 80 I=1,N1
   K0=J7+I

```


AMRK10340
 AMRK10500
 AMRK10600
 AMRK10700
 AMRK11000
 AMRK11100
 AMRK11200
 AMRK11300
 AMRK11400
 AMRK11500
 AMRK11600
 AMRK11700
 AMRK12000
 AMRK12100
 AMRK12200
 AMRK12300
 AMRK12400
 AMRK12500
 AMRK12600
 AMRK12700
 AMRK13000
 AMRK13100
 AMRK13200
 AMRK13300
 AMRK13400
 AMRK13500
 AMRK13600
 AMRK13700
 AMRK14000
 AMRK14100
 AMRK14200
 AMRK14300
 AMRK14400
 AMRK14500
 AMRK14600
 AMRK14700
 AMRK15000
 AMRK15100
 AMRK15200
 AMRK15300
 AMRK15400
 AMRK15500
 AMRK15600
 AMRK15700
 AMRK16000
 AMRK16100
 AMRK16200

```

80 T(K0)=C(1)*D(I+1)
C
C
C   COMPUTE K1
DELT=C*.5*C(1)
TME=DELT+TME
V(1)=TME
DO 90 I=1,N1
  K0=J7+I
  K1=J11+I
  K2=J12+I
  K3=J13+I
  K5=J15+I
  T(I)=C(I+1)
  T(K1)=T(K2)
  T(K2)=T(K3)
  T(K3)=T(K5)
  T(K5)=T(K5) +0.5*T(K0)
  V(I+1)=T(K5)
  CALL ALXSLB
  DO 100 I=1,N1
    K1=J8+I
    K1=T(K1)=C(1)*D(I+1)
C
C
C   COMPUTE K2
DO 110 I=1,N1
  K5=J15+I
  K1=J8+I
  K0=J7+I
  T(K5)=T(K5) +0.5*(T(K1)-T(K0))
  V(I+1)=T(K5)
  CALL ALXSLB
  DO 120 I=1,N1
    K2=J9+I
    K2=T(K2)=C(1)*D(I+1)
C
C
C   COMPUTE K3
TME=DELT+TME
V(1)=TME
DO 130 I=1,N1
  K5=J15+I
  K2=J9+I
  K1=J8+I
  T(K5)=T(K5) +T(K2)-0.5*T(K1)
  V(I+1)=T(K5)
  CALL ALXSLB
130

```



```

C
C
C
K5=J15+I
COMPUTE Y-PREDICTED.
210
T(K0)=T(K5)
T(K5)=T(K0)+D(1)*(P1*T(I)-P2*T(K4)+P3*T(K1)-P4*T(K2))
V(I+1)=T(K5)
TIME=TIME+C(1)
V(1)=TIME
CALL ALXSLB
K5=0
DO 220 I=1,N1
K0=J8+I
K2=J2+I
K1=J7+I
K4=J1+I
K3=J15+I
C
C
COMPUTE Y-CORRECTED
T(K0)=T(K1)+D(1)*(P4*D(I+1)+C2*T(I)-C3*T(K4)+T(K2)*C4)
ERF=14.*C*MAX1(DABS(T(K1)),.01D0)
ERU=ERF*EL(I)
ERL=ERF*EL(I)
TEMP = CABS(T(K0)-T(K2))
IF (TEMP*.LT.ERU) GO TO 215
IF (TEMP*.ERU) 215,214,214
IF (ABS(SNGL(DELT))*.GT.HMIN) GO TO 260
IF (ABS(SNGL(DELT)) - HMIN) 215,260,260
214 IF(AES(LT.ERL) K5=K5+1
215 CCNT=CCNT+1
IF (TEMP*.LT.ERL) K5=K5+1
IF (TEMP*.ERL) 216,220,220
216 K5=K5+1
220 CONTINUE
IF(K5*.LT.N1) GO TO 300
IF(K5 - N1) 300,221,221
IF(ABS(D(1) + D(1))*.GT.HMAX) GO TO 300
221 IF(AES(D(1) + D(1)) - HMAX) 222,222,300
C
C
C
SET-UP FOR DOUBLING STEP SIZE
IF(KCOUNT*.LE.6) GO TO 300
IF(KCOUNT - 6) 300,300,223
223 CCNT=CCNT+1
223 DO 240 I=1,N1
K1=J1+I
K2=J2+I
K3=J3+I

```

```

AMRK243C
AMRK2440
AMRK245C
AMRK2460
AMRK2470
AMRK250C
AMRK2510
AMRK2520
AMRK2530
AMRK2540
AMRK255C
AMRK2560
AMRK2570
AMRK260C
AMRK2610
AMRK2620
AMRK2630
AMRK2640
AMRK265C
AMRK2660
AMRK272C
AMRK273C
AMRK274C
AMRK277C
AMRK3000
AMRK3010
AMRK302C
AMRK303C
AMRK304C
AMRK305C
AMRK3060
AMRK307C
AMRK310C
AMRK3110
AMRK312C
AMRK3130
AMRK3140
AMRK315C
AMRK3160

```



```

240      K5=J5+I
      T(I)=T(K1)
      T(K1)=T(K2)
      T(K2)=T(K3)
      C(I)=C(I)+C(I)
      KOUNT=4
      DELT=.5*C(I)
      GO TC 300
C
C      SET-UP FOR HALVING STEP SIZE.
C
C 260      CONTINUE
      IF(KCUNT .LE. 4) GO TO 350
      IF(KCUNT - 4) 350,350,261
      TME=TIME-C(I)
261      V(I)=TME
      D(I)=DELT
      DELT=.5*C(I)
      DO 265 I=1,N1
      K0=J7+I
      K1=J1+I
      K2=J2+I
      K3=J3+I
      K5=J15+I
      T(K5)=T(KC)
      V(I+1)=T(KC)
      T(K3)=T(K2)+0.5*(T(K1)-T(K2))
      T(K2)=T(K1)
      T(K1)=T(K1)+0.5*(T(I)-T(K1))
      KOUNT=4
      GO TC 200
C
C      INTEGRATION IS FINISHED. SET UP DERIVATIVES AND EXIT.
C
C 300      DO 310 I=1,N1
      K5=J15+I
      K0=J8+I
      T(K5)=T(K0)
      V(I+1)=T(KC)
      GO TC 170
310      CCNTINUE
C
C      RETURN TO 3RD RK INTEGRATION AND RESTART
C
C 360      DO 360 I=1,N1
      K5=J15+I
      K1=J11+I
      T(K5)=T(K1)

```

```

AMRK3170
AMRK3200
AMRK3210
AMRK3220
AMRK3230
AMRK3240
AMRK3250
AMRK3260
AMRK3270
AMRK3300
AMRK3310
AMRK3320
AMRK3330
AMRK3340
AMRK3350
AMRK3360
AMRK3370
AMRK3400
AMRK3410
AMRK3420
AMRK3430
AMRK3440
AMRK3450
AMRK3460
AMRK3470
AMRK3500
AMRK3510
AMRK3520
AMRK3530
AMRK3540
AMRK3550
AMRK3560
AMRK3570
AMRK3600
AMRK3610
AMRK3620
AMRK3630
AMRK3640
AMRK3650
AMRK3660
AMRK3670
AMRK3700
AMRK3710
AMRK3720
AMRK3730
AMRK3740
AMRK3750
AMRK3760

```


AMRK377C
 AMRK4000
 AMRK401C
 AMRK402C
 AMRK403C
 AMRK404C
 AMRK4050

```

36C  V(I+1)=T(K1)
      TME=TIME-C(1)*4.
      V(1)=TME
      D(1)=CELT
      CALL AUXSUB
      GO TC 50
      END

```

CC

```

SUBROUTINE PROCES
  RETURN
  END

```

CC

```

SUBROUTINE RESET
  COMMON C(3415)
  EQUIVALENCE (C(2300),NOLIST), (C(2301),LISTNC(1)), (C(2351),
1  VALUE(1))
  EQUIVALENCE (C(2675),KRUN)
  DIMENSION LISTNG(50), VALUE(50)
  IF(NOLIST.EQ.0) RETURN
  DO 1 I=1,NCLIST
    J=LISTNC(I)
1  C(J)=VALUE(I)
    KRUN=KRUN+1
  RETURN
  END

```

C

CC C

```

SUBROUTINE INPT1
  COMMON C(3415)
  COMMON/CRCS/ ENG(30),IENG(15)
  NAMELIST /MCNT/ ENG,IENG
  EQUIVALENCE (C(2675), KRUN)
  FLG1=0.C
  IF (KRUN.GT.0) GO TC 100
  READ (5,MCNT)
  WRITE (6,MCNT)
  RETURN
10C END

```

INPT0010

F 56

F 106
 F 108

CC

```

SUBROUTINE INPT2

```

INPT001C


```

C      RETURN
C      END

C      SUBROUTINE INPT3
C      RETURN
C      END

C      SUBROUTINE CUPT1
C      RETURN
C      END

C      SUBROUTINE DUPT2
C      OUTPUT INITIALIZATION SUBROUTINE DUPT2
C      COMMON C(2415)
C      INTEGER PGCNT, DTGNT
C      EQUIVALENCE (C(2660), DTGNT)
C      EQUIVALENCE (C(2661), NCGNT)
C      EQUIVALENCE (C(2662), PGCNT)
C      EQUIVALENCE (C(2667), ITCNT), (C(2668), PCNT), (C(2669), CPP)
C      EQUIVALENCE (C(2670), ITAPE), (C(2671), TAPEND)
C      EQUIVALENCE (C(2909), DCC)
C      EQUIVALENCE (C(2910), KCCNV)
C      KCCNV = C
C      ITCNT = DCC + 1.0
C      PGCNT = C.C
C      DTGNT = 1
C      DTGNT = (NCGNT + 4)/5
C      IF (ITCNT .GE. 7) GO TO 2
C      IF (ITCNT - 7) 1, 2, 2
C      1 WRITE (6, 6) (I, C(I), C(I+1), C(I+2), C(I+3), C(I+4), C(I+5), C(I+6),
C      2 C(I+7), I=1, 3415, 8)
C      6 FORMAT(1P1/(15, 2X, 8F14.7))
C      TAPEND = C.
C      K = TAPEND
C      IF (K .NE. 0) REWIND K
C      IF (K) 3, 4, 3
C      3 RETURN
C      4 END

C      C
C      C
C      C

```


CLPT002C
CLPT001C
CLPT017C
CLPT003C
CLPT020C

CLPT007C
CLPT010C
CLPT011C
CLPT012C
CLPT013C
CLPT014C
CLPT015C
CLPT016C

CLPT021C
CLPT022C
CLPT023C
CLPT024C
CLPT025C
CLPT026C
CLPT027C

CLPT034C
CLPT035C
CLPT036C

CLPT040C
CLPT041C

```

SUBROUTINE CLPT3
  CUTPLT SUBROUTINE CUFT3
  DIMENSION B(50), OUTNO(50), ONAME1(50), ONAME2(50)
  DIMENSION CNAME3(50)
  COMMON C(3415)
  INTEGER ITCNT, PGCNT, CUTNO
  EQUIVALENCE (C(2500), CUTNO(1))
  EQUIVALENCE (C(2550), CNAME1(1))
  EQUIVALENCE (C(2600), CNAME2(1))
  EQUIVALENCE (C(2150), CNAME3(1))
  EQUIVALENCE (C(2660), DTCNT)
  EQUIVALENCE (C(2661), NCOU)
  EQUIVALENCE (C(2662), PGCNT)
  EQUIVALENCE (C(2667), ITCNT), (C(2668), PCNT), (C(2669), CPP)
  EQUIVALENCE (C(932), I)
  EQUIVALENCE (C(2913), DER)
  EQUIVALENCE (C(2670), TAPE), (C(2671), TAPEND)
  EQUIVALENCE (C(0290), XTP), (C(0298), YTP), (C(0306), ZTP)
  EQUIVALENCE (C(3400), PUNCH)
  EQUIVALENCE (C(0521), THET)
  EQUIVALENCE (C(0522), PSI)
  EQUIVALENCE (C(0523), PHI)
  EQUIVALENCE (C(0554), AXECI)
  EQUIVALENCE (C(0555), AYECI)
  EQUIVALENCE (C(0556), AZECI)
  EQUIVALENCE (C(0577), VXECI)
  EQUIVALENCE (C(0552), VYECI)
  EQUIVALENCE (C(0553), VZECI)
  EQUIVALENCE (C(0547), XECI)
  EQUIVALENCE (C(0548), YECI)
  EQUIVALENCE (C(0549), ZECI)
  IF (ITCNT - GT. 6) GO TO 7
  IF (ITCNT - 6) I = 10, 7
  1C ITCNT = ITCNT + 1
  WRITE (6, 6) I, C(I), C(I+1), C(I+2), C(I+3), C(I+4), C(I+5), C(I+6),
    1 C(I+7), I = 1, 3415, 8)
  6 FORMAT(1H1/(15, 2X, 8E14.7))
  PGCNT = 1
  7 GCNT INCL
  8 IF (I - LT. PCNT) RETURN
  9 IF (I - PCNT) 12, 9, 9
  12 RETURN
  PCNT = PCNT + CPP
  9 PCNT = PGCNT + NE. 1) GC TC 3
  IF (PGCNT - 1) 3, 1, 2
  IF (PGCNT - 1) 3, 1, 2
  1 WRITE(6, 2) (ONAME1(I), ONAME2(I), ONAME3(I), I = 1, NCOU)

```

C

C

C

C
C
C

STGE0640
 STGE0650
 STGE0660
 STGE0670
 STGE0700
 STGE0710
 STGE0720
 STGE0730
 STGE0740
 STGE0750
 STGE0760
 STGE0770
 STGE1000
 STGE1010

CNTR0010
 CNTR0020
 CNTR0030

CNTR0010
 CNTR0020
 CNTR0030

CNTR0010
 CNTR0020
 CNTR0030

RNDM0010
 RNDM0020
 RNDM0030

IF (K .NE. 0) FND FILE K
 IF (K .NE. 0) REWIND K
 IF (K) 16,17,16
 16 END FILE K
 17 REWIND K
 17 KSTEF = 2
 17 IF (LCSTAT .EQ. NOSTAT) GO TO 4
 17 IF (LCSTAT - NOSTAT) 18,4,18
 18 L = LCSTAT + 1
 18 DO 3 I = L, NOSTAT
 18 RMS(I) = C(N)
 18 RETURN
 18 = 4
 18 END

SUBROUTINE CNTR1
 RETURN
 END

SUBROUTINE CNTR2
 RETURN
 END

SUBROUTINE CNTR3
 RETURN
 END

SUBROUTINE RNDM1
 RETURN
 END

SUBROUTINE RNDM2

COMMON C(3415)
 EQUIVALENCE (C(2443), RNFLG)
 EQUIVALENCE (C(2446), ZN)
 EQUIVALENCE (C(2448), NCFNDM)
 EQUIVALENCE (C(2449), CELT)
 EQUIVALENCE (C(2450), RNDMNO)
 EQUIVALENCE (C(2913), CER)
 DIMENSION RNDMNO(50)
 INTEGER RNDMNO


```

3 IF (ACRNDM) 2,2,3
  RNFLG=C
  DELT=DER
  DO 1 I=1,ACRNDM
    J=RNLMNC(I)
    C(J+5)=2.718281828*(-DER*C(J+4))
    C(J+6)=C(J+3)*SQRT(1.0-C(J+5)*C(J+5))
    ZN=C(J+1)
    Y=C(J)
    N=INT(Y)
    CALL RFG(N,X)
    C(J+7)=C(J+3)*X
    C(J+1)=ZN
  1 RETURN
  2 END

```

C C C

```

SUBROUTINE RADM3
COMMON C(3415) (2443), RNFLG)
EQUIVALENCE (C(2444), RN)
EQUIVALENCE (C(2446), ZN)
EQUIVALENCE (C(2448), NORNDM)
EQUIVALENCE (C(2449), DELT)
EQUIVALENCE (C(2450), RADMNC)
EQUIVALENCE (C(2913), DER)
DIMENSION RADMNO(50)
INTEGER RADMNO
IF (ACRNDM) 20,20,10
IF (RNFLG-1.0) GO TO 8
IF (DER-0.0) GO TO 6
DO 9 I=1,ACRNDM
  J=RNLMNC(I)
  X=(C(J+7)-C(J+5)*C(J+8))/C(J+6)
  IF (RNFLG-1.0) GO TO 5
  RNFLG=1.0
  DELT=DER
  GO TO 7
DELT=DELT+DER
C(J+5)=2.718281828*(-DELT*C(J+4))
C(J+6)=C(J+3)*SQRT(1.0-C(J+5)*C(J+5))
C(J+7)=C(J+6)*X+C(J+8)
RETURN
IF (DELT-0.0) GO TO 4
DO 3 I=1,ACRNDM
  J=RNLMNC(I)

```



```

3 C(J+5)=2.718281828*(-DER*C(J+4))
  C(J+6)=C(J+3)*SQRT(1.0-C(J+5)*C(J+5))
4 DELT=DER
  DO 1 I=1,NCRNDM
    J=RNCMNC(I)
    Y=C(J)
    ZN=C(J+1)
    N=INT(Y)
    CALL RFG(N,X)
    C(J+8)=C(J+7)
    C(J+7)=C(J+6)*X+C(J+5)*C(J+7)
    C(J+1)=ZN
1  C(J+1)=ZN
20 RETURN
  END

```

C C

```

SUBROUTINE AUXA1
RETURN
END

```

C C

```

SUBROUTINE AUXA2
RETURN
END

```

C C

```

SUBROUTINE AUXA3
RETURN
END

```

C C

```

SUBROUTINE AUXB1
RETURN
END

```

C C

```

SUBROUTINE AUXB2
RETURN
END

```

C C

```

SUBROUTINE AUXB3
RETURN
END

```

C C

```

SUBROUTINE AUXC1
RETURN

```

ALXA0010
ALXA002C
ALXA003C

ALXB0010
ALXB002C
ALXB0030

ALXB001C
ALXB002C
ALXB003C

ALXB001C
ALXB002C
ALXB003C

ALXC001C
ALXC002C

ALXC0020
ALXC0010
ALXC0020
ALXC0030
ALXC0010
ALXC0020
ALXC0030

```

END
SUBROUTINE ALXC2
RETURN
END

SUBROUTINE AUXC3
RETURN
END

SUBROUTINE RFG(N,X)
COMMON C(2415)
EQUIVALENCE (C(2446), ZN)
Y=FLCAT(N)
X=0.0
GENERATE UNIFORM RANDCN NO.(0 TO 1)
SUM FCR RAND NO(SIGMA=1,MEAN=0)
UNIFORM CIST FOR N=1
NORMAL CIST FOR LARGE N (12 OR GREATER)
DO 1 I=1,N
X=X+ZN-0.5
RNC=SGN(*ZN)
ZN=RNDA-ABNT(RNO)
X=SGRT(12./Y)*X
RETURN
END
1

```

```

SUBROUTINE AII
AERODYNAMIC FORCES AND MOMENTS INITIALIZATION MODULE AII
BODY AXES
COMMON C(2415)
EQUIVALENCE (C(0208), TSA )
EQUIVALENCE (C(0257), CTS )
EQUIVALENCE (C(0258), STS )
CTS = CCS(TSA)
STS = SIN(TSA)
RETURN
END

```


[illegible]

```

CAB - ZERC-LIFT BODY DRAG
CAC - DRAG DUE TO DEFLECTED CONTROL SURFACES
CAE+CAC
CA - CARMAL FORCE DUE TO ANGLE OF ATTACK
CZC - NCRMAL FORCE DUE TO FIN DEFLECTION
CZ - NORMAL FORCE DUE TO FIN DEFLECTION
CYB - SIDE FORCE DUE TO SIDESLIP
CYC - SIDE FORCE DUE TO FIN DEFLECTION
CY - SIDE FORCE
CMG - PITCH DAMPING MOMENT
CNR - YAW DAMPING MOMENT
CLP - ROLL DAMPING MOMENT
CLB - PITCH MOMENT DUE TO ANGLE OF ATTACK
CMC - CCNTRCL SURFACE PITCH MOMENT
CM - CME+CNC
CNB - YAW MOMENT DUE TO SIDESLIP
CNC - CCNTRCL SURFACE YAW MOMENT
CN - CNE+CNC
CLC - INDUCED ROLL MOMENT
CCL - CCNTRCL SURFACE ROLL MOMENT
CCLC - CLB+CCL

COMMON/APPLE/TM1(7), TM2(15), TM3(14), TM4(5), TETA1(7), TETA2(7),
1 TETA3(6), TX11(2), TX12(10), TX13(5), TDELTA(4), TH1(3), TH2(2),
2 TCZBC(7,2), TDCZP(14,7), TCYBC(7,10,6), TCM8C(7,5,7), TDCMF(14,7),
3 TCCEB(15,2), TCDC(15,2), TCAD2(15), TCN8C(7,10,6), TCM01(15,4),
4 TCMD2(5,4), TCCMD(15), TCCMCA(15), TPMD(14,3), TCMQ(15), TPQ(14,3),
5 TCCLBCC(7,10,6), TCCLD(15), TPLD(14,3), TCLP(15), TPP(14,3), TDCYSP(3),
L TDCZSP(7,3), TDCMSP(3), TFCNSP(3), TH3(3), TFEJ(13), TFEJ(13)
COMMON C(3415)
EQUIVALENCE C(0016), DPI
EQUIVALENCE C(0020), CG
EQUIVALENCE C(0024), CR
EQUIVALENCE C(0111), CA
EQUIVALENCE C(0113), CY
EQUIVALENCE C(0114), XI
EQUIVALENCE C(0115), CZ
EQUIVALENCE C(0116), CBRF
EQUIVALENCE C(0117), FTA
EQUIVALENCE C(0118), CMG
EQUIVALENCE C(0119), CNR
EQUIVALENCE C(0120), CLP

```



```

VN=SCRT(VEA**2+WBA**2)
IF (VN.LT.C.00001) GC TC 10
XIR=ATAN2(VBA,WBA)
GO TC 20
1C XIR=C.C
20 XI=XIR*RAD
XI1=XI
IF (XI.LT.C.C) XI1=-XI
XI2=XI1
IF (XI1.GE.90.) XI2=180.-XI1
C TABLE LOCK-UP,ALL PHASES
CCZP=0.0
CCMP=C.C
PMC=1.0
FNC=1.0
PLD=1.0
PQ=1.0
PR=1.0
CCNSEP=C.
CCNSEP=C.
CCNSEP=C.
CCYSEP=C.
CCZSEP=C.
CCLSP=C.
CCYSP=C.
CCMENT CFFICIENTS
FITTING MMENT
CMBC=THREDL(AM,XI2,ETA,TM1,TXI3,TETA1,TCMBC,7,5,7)
CCMUL=CLLI(AM,TM2,TCMD,15)
CCMUR=CCMLL
CCMLL=-CCMUL
CCMLR=-CCMUL
CCMLA=CLLI(AM,TM2,TCMDA,15)
CCMURA=-CCMULA
CCMLLA=-CCMULA
CCMLRA=CCMLA
CCMG=CLLI(AM,TM2,TCMG,15)
YAWING MMENT
CNBC=THREDL(AM,XI2,ETA,TM1,TXI2,TETA3,TCNBC,7,10,6)
IF (XI1.GT.90.) CNBC=-CNBC
IF (XI.LT.0.0) CNBC=-CNBC
CCNUL=-CCMUL
CCNULF=CCMUL
CCNLL=-CCMUL
CCNLR=CCMUL
CCNULB=CCMULA

```



```

DCNURB=-DCMULA
DCNLLB=-DCMULA
DCNLRB=CCMULA
CNR=CMG
RCLLIANG NCMENT
CLBC=TFRECL(AM,XI2,ETA,TM1,TXI2,TETA3,TCLBC,7,10,6)
IF (XI1.GT.90.) CLBC=-CLBC
IF (XI1.LT.0.) CLBC=-CLBC
CLD=CDLI(AM,TM2,TCLD,15)
CLF=CDLI(AM,TM2,TCLF,15)
FCRCE CCEFFICIENTS
CYBCC=TFRECL(AM,XI2,ETA,TM1,TXI2,TETA1,TCZBC,7,2,7)
CYBC=TFRECL(AM,XI2,ETA,TM1,TXI2,TETA3,TCYBC,7,10,6)
IF (XI1.GT.50.) CYBC=-CYBC
IF (XI1.LT.0.) CYBC=-CYBC
CAD2=CDLI(AM,TM2,TCAD2,15)
IF (1.GE.13) GO TO 100
IF (1.LT.13) SEP AND BOOST PHASE
TABLE LOCK-LF,
ZERC-LIFT CRAG
CDO=STCLIA(TM2,TH2,TCDB,AM,H,15,2)
IF (1.LT.11) CDO=CCC+.08
CCTRCL SURFACE EFFECTIVENESS
CMULC=STCLIA(TM2,TDELT,TCMD1,AM,ADUL,15,4)
IF (DCL.LT.0.0) CMULC=-CMULC
CMURC=STCLIA(TM2,TDELT,TCMD1,AM,ADUR,15,4)
IF (CUR.GT.0.0) CMURC=-CMURC
CMLLC=STDLIA(TM2,TDELT,TCMD1,AM,ADLL,15,4)
IF (CLL.LT.0.0) CMLLC=-CMLLC
CMLRC=STDLIA(TM2,TDELT,TCMD1,AM,ADLR,15,4)
IF (CLR.GT.0.0) CMLRC=-CMLRC
FLUME EFFECTS
IF (AM.GE.1.6) GO TO 200
FOK=F/ICCC.
FM=2.1212E-06*HOK**3-5.2017E-04*HOK**2+.044193*HOK-9.0909E-04
DCMP=STCLIA(TM3,TETA2,TDCMP,AM,ETA,14,7)
IF (DCMP.LT.0.0) DCMP=C.0
CCZP=STCLIA(TM3,TETA2,TCCZP,AM,ETA,14,7)
IF (DCZP.GT.0.0) DCZP=C.0
CCMP=CCMP*FM
DCZF=DCZF*FM
PMD=STCLIA(TM3,TH1,TFMD,AM,H,14,3)
IF (PMD.GT.1.0) PMD=1.0
FND=FM
PLD=STCLIA(TM3,TH1,TFLD,AM,H,14,3)
IF (PLD.GT.1.0) PLD=1.0
FQ=STCLIA(TM3,TH1,TFQ,AM,H,14,3)
IF (FQ.GT.1.0) PQ=1.0
PR=PC

```



```

PP=STCLIA(TM3,TH1,TFF,AM,H,14,3)
IF (FP,GT,1.0) PP=1.0
SEPARATE EFFECTS
IF (FLGSEP,LE,0.) GO TC 200
IF (T,GE,1.0) GO TC 200
IF (ZTF,GE,15.) GO TC 200
FS2=(1.-ZTF/15.)*EXP(-ZTF/15.)
DCZSFP=CCLI(H,TH3,TCCNSP,3)
DCNSF=FS2*DCNSPP
DCZSF=FS2*DCZSPP
IF (ZTF,GE,4.0) GO TC 30
FS1=(1.-ZTF/4.)*EXP(-ZTF/4.)
DCYSFP=CCLI(H,TH3,TCCYSP,3)
DCNSFP=FS1*DCNSPP
DCYSF=FS1*DCYSPP
DCZSF=FS1*DCZSPP
CPHI=CCS(PHI)
SPHI=SIN(PHI)
LCZSEP=LCZSP*CPHI+DCYSP*SPHI
DCYSEP=DCYSP*CPHI-DCZSP*SPHI
DCMSEP=DCMSP*CPHI+DCNSP*SPHI
DCNSEP=DCNSP*CPHI-DCMSP*SPHI
GO TC 200
LF,CRUISE PHASE
TABLE LOCK DRAG
ZERC=LIFT CLIA(TM2,TH2,TCC,AM,H,15,2)
100 CCMUL=STCLIA(TM4,TDELT,TCMD2,AM,ADUL,5,4)
IF (DUL,LT,0.) CMLC=-CMLD
CMURC=STCLIA(TM4,TDELT,TCMD2,AM,ADUR,5,4)
IF (DUR,GT,0.) CMUR=-CMURD
CMLC=STCLIA(TM4,TDELT,TCMD2,AM,ADLL,5,4)
IF (DLL,LT,0.) CMLC=-CMLD
CMLRC=STCLIA(TM4,TDELT,TCMD2,AM,ADLR,5,4)
IF (DLR,GT,0.) CMLRC=-CMLRD
CLE=CLE+0.017
END CF TABLE LOCK-UP
200 CCMULC=CMLC
CMURC=-CMLC
CMLRC=-CMLRC
CZEC=CZBCC+DCZP
CMFC=CMFCC+CCMP
CABC=CLC*CCSETA
C TRANSFORM FROM CROSS-FLW TO BODY AXIS COMPONENTS

```



```

CXI=CCS(XIR)
SXI=SSIN(XIR)
CAE=CAEC
CYE=CYEC
CLE=CLBC
CNB=CNBC
CNTRCL=CNB
CNTRCL=CNB
F=(15.25-CCR)/(15.25-CCR)
CMUL=CNULC+DCMURA*ALF*DLR+DCMUR*ADUL
CMLL=CNLLC+DCMLLA*ALP*DLR+DCMLL*ADLL
CMLR=CNMLC+DCMLRA*ALP*DLR+DCMLR*ADLR
CMC=(CNULC+CMUR+CNLL+CNLR)*F*PMD
CNUL=CNULC+CNULB*BF*DLR+DCNUL*ADUL
CNUR=CNURC+CNURB*BF*DLR+DCNUR*ADUR
CNLL=CNLLC+CNLLB*BF*DLR+DCNLL*ADLL
CNLR=CNLRC+CNLRB*BF*DLR+DCNLR*ADLR
CNC=(CNULC+CNUR+CNLL+CNLR)*F*PND
CLUL=CLLC*DLR
CLUR=CLLC*DLR
CLLL=CLLD*DLR
CLLR=CLLD*DLR
CLC=(CLLL+CLUR+CLLL+CLLR)*PLD
CZC=-CNC*CBAR/(15.25-CCR)
CYC=-CNC*CBAR/(15.25-CCR)
CAC=CAC2*(DLR**2+DLR**2+CLL**2+DLR**2+4.*1.3*(ALP*DELQ-BET*DELQ))
TOTAL AERCDYNAMIC FORCE COEFFICIENTS
CA=CAE+CAC
CY=CYE+CYC+CCYSEP
CZ=CZB+CZC+CCZSEP
CA=-CA
CZ=-CZ
TOTAL AERCDYNAMIC MOMENT COEFFICIENTS
CLE=CLE+CLC+CCLEP
CLF=CLF*FAC*PP
CM=CMB+CMC+CZB*(CG-CCR)/CBAR+DCMSEP
CMG=CMG*FAC*PP
CNR=CNR+CMC+CYB*(CG-CCR)/CBAR+DCNSEP
RETURN
END

```



```

SUBROUTINE A3I
COMMON C(3415)
COMMON /FLAG/ FLG1
EQUIVALENCE (C(0601), FE1)
EQUIVALENCE (C(0602), WEIG)
EQUIVALENCE (C(0603), IA)
EQUIVALENCE (C(0604), FLG2)
FE1=C.C
WEIG=C.C
IA=0
FLG1=C.C
FLG2=C.C
RETURN
END

```

CCCCCCCC

SUBROUTINE A3

THIS IS THE MISSILE PROPUSSION MODULE

```

T1=BOOST ENGINE IGNITION TIME
BT=BOOST ENGINE BURNCUT TIME(=T3)
BE=CRUISE ENGINE IGNITION TIME(=T4)

```

```

COMMON /FLAG/ FLG1
COMMON C(3415)
EQUIVALENCE (C(0086), S)
EQUIVALENCE (C(0110), T)
EQUIVALENCE (C(0932), TI)
EQUIVALENCE (C(0933), TT)
EQUIVALENCE (C(0935), BT)
EQUIVALENCE (C(0936), BE)
EQUIVALENCE (C(0136), CG)
EQUIVALENCE (C(0150), OLT)
EQUIVALENCE (C(0151), CMT)
EQUIVALENCE (C(0152), CNT)
EQUIVALENCE (C(0153), FPSTH)
EQUIVALENCE (C(0154), PHITH)
EQUIVALENCE (C(0201), A)
EQUIVALENCE (C(0202), B)
EQUIVALENCE (C(0203), CC)
EQUIVALENCE (C(0073), TXBA)
EQUIVALENCE (C(0074), TYEA)
EQUIVALENCE (C(0075), TZE)
EQUIVALENCE (C(0507), F)
EQUIVALENCE (C(0520), AMACH)
EQUIVALENCE (C(0117), ETA)

```



```

EQUIVALENCE (C(0601), FF1)
EQUIVALENCE (C(0602), WEIG)
EQUIVALENCE (C(0603), IA)
EQUIVALENCE (C(0604), FLG2)
EQUIVALENCE (C(0605), DIFFM)
EQUIVALENCE (C(0606), FLGRJ)
EQUIVALENCE (C(0508), GD)
EQUIVALENCE (C(2913), DER)
DATA WFLEL/189.7

C 80 FORMAT(1H,10X,'TOTAL ANGLE OF ATTACK EXCEEDED, ETA=',F8.3)
C
IF (FLG1.GT.0.0) GO TO 1C
CT = 0.0
DIFFM = C
FF = 0.0
TXBA = C.0
TYBA = C.0
TZBA = 0.0
IF (T.LT.11) RETURN
FIN = F*12.*.0254
TIN = T
AIN = ETA
IF (T.GE.BE) GO TO 1
C BCCSTT = T
IF (T.GT.ET) GO TO 10
CALL BCCSTT(TT,THRU,FF)
TXBA = THRU*CS(CPS*TH)
TYBA = THRU*SS(CPS*TH)
TZBA = TNEA*CS(PHIT)
IB = 0
WEIGHT = WEIGHT-FF+FF1
CG = CG - (FF-FF1)/405.94
A = A - (FF-FF1)/256.25
B = B - (FF-FF1)/1.881
CC = B
CLT = C
CMT = TZEA*(15.525-CG)
CNT = -TYEA*(15.525-CG)
FF1 = FF
TW = T+DER
IF (TW.LT.ET) GO TO 2
IF (FLG2.GT.0.0) GO TO 2
WEIG = CG + .15
CG = CG - .15
FLG2 = 1.0

```



```

10 GO TC 2
CONTINUE
IF (FLGRJ.GT.0.0) GO TO 21
CRUISE PHASE, RAMJET CFF
IF (IB.GE.1) GO TO 11
CALL ENGINE (HIN, AMACH, AIN, 0., TIN, 1.0, 0.0, 0., CT, SMARG, -1, 1)
IB=1
11 IF (AIN.GT.5.3) AIN=5.3
CALL ENGINE (HIN, AMACH, AIN, 0., TIN, 1.0, 0.0, 0., CT, SMARG, 1, 1)
GO TC 4
C SIMPLIFIED RAMJET MODEL
C CRUISE PHASE, RAMJET CFF
21 CALL RAMJET (HIN, AMACH, AIN, FF, 1.0, CT)
GO TC 4
1 CONTINUE
IF (FLGRJ.GT.0.0) GO TC 22
CRUISE PHASE, RAMJET CN
IB=0
IF (AIN.GT.10.0) GO TC 5
FF=FF1
IF (IA.GE.1) GO TC 3
CALL ENGINE (HIN, AMACH, AIN, FF, TIN, 0.0, 0.0, 0.0, CT, SMARG, -1, 1)
TT=1
FF1=FF
IA=1
3 XIN=0.0
IG.GT.WFUEL) XIN=1.0
IF (WEENGINE (HIN, AMACH, AIN, FF, TIN, XIN, 0.0, 0.0, CT, SMARG, 1, 1)
CALL TC 4
C SIMPLIFIED RAMJET MODEL
C CRUISE PHASE, RAMJET CN
22 IF (AIN.GT.10.0) GO TC 5
XIN=0.0
IG.GT.WFUEL) XIN=1.0
CALL WEENGINE (HIN, AMACH, AIN, FF, TIN, XIN, CT)
GO TC 4
5 IF (FLG1.GT.0.0) GO TC 10
WRITE (C, EC) ETA
FLG1=1.0
GO TC 10
TXBA=CT*GD*S
TYEA=0.
TZBA=CT
CT=T-TT
IF (FLGRJ.GT.0.0) GO TC 6
FF=FF/.45359237
C DIFFUSER (INLET) MARGIN IS NOT COMPUTED IN SIMPLIFIED RAMJET MODEL
C DIFFM=SMARG/100.

```



```

C WEIG=WEIG+(ABS(FF+FF1))*CT/2.0
C IF (WEIG.GT.WFUEL) FLG1=1.0
C WEIGHT = WEIGHT - ABS(FF + FF1)*DT/2.
C TT=TT
C FF1 = FF
C CG=CG-.00026455*ABS(FF+FF1)*DT/2.
C A=A-.0031746*ABS(FF+FF1)*DT/2.
C B=B-.063492*ABS(FF+FF1)*CT/2.
C CC=B
C RETURN
C END

```

2

CC C
CC C

SUBROUTINE A4I

ACTUATOR INITIALIZATION MODULE A4I

```

COMMON C(3415)
DIMENSION IPL(100)
EQUIVALENCE (C(1735),WFIN)
EQUIVALENCE (C(3315),N)
EQUIVALENCE (C(3316),IPL(1))
IF (WFIN.GT.C.0) GO TO 100
GO TO 200
100 IPL(N)=1653
IPL(N+1)=1697
IPL(N+2)=1701
IPL(N+3)=1705
IPL(N+4)=1709
IPL(N+5)=1713
IPL(N+6)=1717
IPL(N+7)=1721
N=N+8
RETURN
END

```

100

200

CC C
CC C

SUBROUTINE A4

FIN ACTUATORS MODULE A4

```

COMMON C(3415)
EQUIVALENCE (C(0016),CP)
EQUIVALENCE (C(0020),CG)
EQUIVALENCE (C(0024),DR)
EQUIVALENCE (C(1693),CULC)

```



```

C C C
C C C
C C C
SUBROUTINE CII
INITIALIZATION MODULE FOR LVRJ AUTOPILOT

COMMON C(3415)
EQUIVALENCE (C(3315),N)
EQUIVALENCE (C(3316),IPL(1))
DIMENSION IPL(100)
IPL(N)=1677
IPL(N+1)=1689
IPL(N+2)=1725
IPL(N+3)=1725
N=N+4
RETURN
END

```

```

C C C
C C C
C C C
SUBROUTINE C1
SUBROUTINE C1-LVRJ
THIS AUTCFILCT MODULE IS FOR STV G

COMMON C(3415)
EQUIVALENCE (C(0581), DYAP)
EQUIVALENCE (C(0515), PSF)
EQUIVALENCE (C(0520), AMACH)
EQUIVALENCE (C(0415), G)
EQUIVALENCE (C(0507), F)
EQUIVALENCE (C(0932), T)
EQUIVALENCE (C(0933), T1)
EQUIVALENCE (C(0935), T3)
EQUIVALENCE (C(1636), ANZC)
EQUIVALENCE (C(1640), ANYC)
EQUIVALENCE (C(1644), AZM)
EQUIVALENCE (C(1648), AYM)
EQUIVALENCE (C(1652), GM)
EQUIVALENCE (C(1656), RM)
EQUIVALENCE (C(0980), PM)
EQUIVALENCE (C(1677), DPCID)
EQUIVALENCE (C(1680), DPCI)
EQUIVALENCE (C(1689), P10)
EQUIVALENCE (C(1692), P1)
EQUIVALENCE (C(1725), Z1C)
EQUIVALENCE (C(1728), Z1)

```



```

EQUIVALENCE (C(1729),Y1D)
EQUIVALENCE (C(1732),Y1)
EQUIVALENCE (C(1737),C6)
EQUIVALENCE (C(1738),DRC)
EQUIVALENCE (C(1739),EPC)
EQUIVALENCE (C(1740),DPA)
EQUIVALENCE (C(0165),AKN)
EQUIVALENCE (C(0166),AKC)
EQUIVALENCE (C(0167),AKY)
EQUIVALENCE (C(0168),AKR)
EQUIVALENCE (C(0169),AKF)
AKG11=7.55
AKR11=7.55
AKP11=10.0
AKR12=3.0
AKS=-2.75/57.2957795
QCE=1.45/57.2957795
IF (DYNP.GT.22.5) GC TC 3
AKY=.1
AKN=.1
IF (DYNP.GT.4.0) GO TC 4
AKF=.8
AKR=1.1
AKC=1.6
GO TC 5
3 AKNY=.3
1 DYNP=.3
IF (DYNP.GT.45.0) GC TC 5
4 IF (DYNP.GT.35.0) GC TC 6
1 AKF=1.0
1 +1.5
AKQ=1.5
AKQ=1.5
1 +5.278795E-06*DYNP**4
GC TC 8
5 AKN=.3
6 AKP=1.752142-.00285476*DYNP+1.47619E-05*DYNP**2
AKR=1.556357-.0665024*DYNP+.00112333*DYNP**2-6.66666E-06*DYNP**3
AKC=1.408333-.0036667*DYNP
8 IF (T.GT.1) GO TC 1
IF (AMACH.GT.2.0) GC TC 1
C SEPARATION AND BOOST AUTCPLOT
GC=CCS
IF (T.CE.T1) QC=QCE
S1=1.0

```



```

S2=0.0
AKGI=AKGI1
AKRI=AKRI1
AKPI=AKPI1
GO TIC 16
C CRUISE ALT FICT
1 CONTINUE
S1=0.0
S2=1.0
AKGI=AKGI2
AKRI=AKRI2
AKPI=AKPI2
16 CONTINUE
FITCH AND YAW CHANNELS
Q1=AKQ*(CM-S1*QC)
R1=AKR*RM
ANZN=-AZM/G
ANYN=AYN/G
ANZ=ANZC-ANZM
ANY=ANYC-ANYM
POS DQC AND DRC COMMAND NEG TURNING MOMENTS
LQCI=Q1-S2*AKNZ*ANZ
ZID=QCI
IF (ABS(LQCI).GE.DMAX) ZID=0.0
LGC=LQCI+AKCI*Z1
IF (ABS(LGC).GF.DMAX) DCC=SIGN(DMAX,DQC)
DRC1=R1-S2*AKNY*ANY
YID=LRC1
IF (ABS(LRC1).GE.DMAX) YID=0.0
IF (LRC1+AKRI*Y1)
IF (ABS(LRC).GF.DMAX) DRC=SIGN(DMAX,DRC)
2 CONTINUE
RCLL CHANNEL
PLC=AKPI*FM
LAG CCMPENSATOR
CMG1=1
CMG2=.01
CPCC=-AKP*(FM+P1)*QMG2/CMG1
LPCIC=(CMG1-CMG2)*EFC(-CMG2*DPCI
DPC=GPCI+DPCD
IF (ABS(DPC).GF.DMAX) DPC=SIGN(DMAX,DPC)
RETURN
END
C
C
SUBROUTINE C2I
C

```



```

DIMENSION IPL(100)
COMMON C(3415),
EQUIVALENCE (C(0507),F)
EQUIVALENCE (C(3315),N)
EQUIVALENCE (C(3316),IPL(1))
IPL(N)=1633
IPL(N+1)=1637
N=N+2
HNCK=H
RETURN
END

```

CC C CC C

SUERCUTINE C2 GLIDANCE CCMAND MODULE

```

COMMON C(3415)
EQUIVALENCE (C(0256), A33)
EQUIVALENCE (C(0286), VXTP)
EQUIVALENCE (C(0294), VYTP)
EQUIVALENCE (C(0302), VZTP)
EQUIVALENCE (C(0337), CMGYC)
EQUIVALENCE (C(0338), CMGZC)
EQUIVALENCE (C(0339), FLGT)
EQUIVALENCE (C(0339), XMA)
EQUIVALENCE (C(0340), YMA)
EQUIVALENCE (C(0341), ZMACH)
EQUIVALENCE (C(0500), FDC)
EQUIVALENCE (C(0501), FREF)
EQUIVALENCE (C(0502), HD)
EQUIVALENCE (C(0507), F)
EQUIVALENCE (C(0508), GC)
EQUIVALENCE (C(0515), PSF)
EQUIVALENCE (C(0527), GAMMAV)
EQUIVALENCE (C(0528), VT)
EQUIVALENCE (C(0531), VF)
EQUIVALENCE (C(0581), CYNPI)
EQUIVALENCE (C(0582), AKG)
EQUIVALENCE (C(0932), T1)
EQUIVALENCE (C(0933), T2)
EQUIVALENCE (C(0935), T3)
EQUIVALENCE (C(0937), T5)
EQUIVALENCE (C(0942), TDC)
EQUIVALENCE (C(0944), TLD)
EQUIVALENCE (C(0945), TFC)

```



```

EQUIVALENCE (C(1633),ANZCD)
EQUIVALENCE (C(1636),ANZC)
EQUIVALENCE (C(1637),ANYCD)
EQUIVALENCE (C(1640),ANYC)
EQUIVALENCE (C(1750),WFI)
DATA RCCN/3E500./
C DYNP=CD/144.
C ZERC CONTROL SWITCHES
S4=0.0
S5=C.C
S6=C.C
S8=0.0
IF (T.GE.T3) GO TC 1
C SEPARATION,BCCST PHASE
IF (ANACT.GE.2.0) GC TC 1
GO TC 100
C ZERC TRAJECTORY
1 IF (T.GE.T5) GO TO 2
GO TC 100
C LEVEL FLIGHT
2 IF (T.GE.TDC) GO TO 5
S4=1.0
S5=1.0
GO TC 100
C DIVE,CLIMB,GE.TLC GO TO 4
5 IF (T.GE.TLC) GO TO 4
7 HDCI=HDC
S4=1.0
S5=1.0
S8=1.0
GO TC 100
C AUTOMATIC LETDCWN TO HREF
4 FORI2=SGRT(XMA**2+YMA**2)
DI=1.575*VF+FCOIN-FCRI2
IF (DI.GE.C.C) GO TC 6
S4=1.0
S5=1.0
S6=1.0
GO TC 100
C TERMINAL FCING
6 W3=(ZMA*VZTP)/(XMA*VXTP+ZMA*VZTP)
ANZC1=-2.*2900.*W3/32.172+A33
IF (FLGT.GT.0.0) GC TC 8
GO TC 100
C TERMINAL FCING
8 ANZC1=AKG*VT*CMGYC/32.174
GO TC 100
100 CCNT INCE

```



```

C *** VERTICAL CHANNEL *****
C HERR LIMITER
  HERR1=SC*(1-1REF)
  HERR2=C.5*HERR1
  IF (ABS(HERR1).GT.8CC.) HERR2=SIGN(400.,HERR1)
  ANZC1=.0333*(S8*HDC1-S4*(HD+HERR2))+S5*A33
C NZ LIMITER
  IF (ANZC1) 108,107,1CE
  IF (ANZC1) 108,107,1CE
108 AZMAX=.35*(DYNP-5.714)
  IF (DYNP.LE.30.0) AZMAX=8.5*DYNP/30.
  AZMAX=AMIN1(AZMAX,12.C)
  IF (ABS(ANZC1).GE.AZMAX) ANZC1=SIGN(AZMAX,ANZC1)
107 CONTINUE
C ***** LATERAL CHANNEL *****
  ANYC1=C
  IF (1.LT.15) GO TO 21C
  IF (FLGT.GT.0.0) GC TC 200
  CROSS-PRCCUCT MIDCOURSE GUIDANCE
  CPROD=(YMA*VXTP-XMA*VYTP)/(XMA*VXTP+YMA*VYTP)
  ANYC1=2900.*CPROD/32.172
  GO TC 205
  TERMINAL FCMING
200 ANYC1=AKG*VT*GMGZC/32.172
C NY LIMITER
205 IF (ANYC1) 208,207,208
208 AYMAX=E.C*(DYNP-11.25)/22.5
  IF (DYNP.LE.22.5) AYMAX=4.0*DYNP/22.5
  AYMAX=AMIN1(AYMAX,12.0)
  IF (ABS(ANYC1).GT.AYMAX) ANYC1=SIGN(AYMAX,ANYC1)
207 CONTINUE
  ELIFTICAL LIMITING
  DENOM=SCRT((AZMAX*ANYC1)**2+(AYMAX*ANZC1)**2)
  IF (DENOM.LE.0.0) GC TO 21C
  AZLIN=AZMAX*AYMAX*ANZC1/CENOM
  IF (ABS(ANZC1).GE.ABS(AZLIN)) ANZC1=AZLIN
  AYLIN=AZMAX*AYMAX*ANYC1/CENOM
  IF (ABS(ANYC1).GE.ABS(AYLIN)) ANYC1=AYLIN
C LATERAL CCM AND FILTER
210 ANYC1=WFL*(ANYC1-ANYC)
  VERTICAL CCM AND FILTER
  ANZC1=WFL*(ANZC1-ANZC)
  RETUEN
  END
C
C
SUBROUTINE C31
RETURN
END

```


CC	SUBROUTINE C3 RETURN END
CC	
CC	SUBROUTINE C4I RETURN END
CC	
CC	SUBROUTINE C4 RETURN END
CC	
CC	SUBROUTINE C5I RETURN END
CC	
CC	SUBROUTINE C5 RETURN END
CC	
CC	SUBROUTINE C6I RETURN END
CC	
CC	SUBROUTINE C6 RETURN END
CC	
CC	SUBROUTINE C7I RETURN END
CC	
CC	SUBROUTINE C7 RETURN END
CC	
CC	SUBROUTINE C8I

RETURN
END

SUBRCUTINE C8
RETURN
END

SUBRCUTINE C9I
RETURN
END

SUBRCUTINE C9
RETURN
END

SUBRCUTINE C10I
RETURN
END

SUBRCUTINE C10
RETURN
END

SUBRCUTINE C11

TRANSLATIONAL DYNAMICS INITIALIZATION MODULE DIAB
FOR USE WITH MODULE CIA CR DIB

DIMENSION IPL (100)
COMMON C(3415)
EQUIVALENCE (C(3315), N,)
EQUIVALENCE (C(3316), IPL(1))
IPL(N) = 283
IPL(N+1) = 291
IPL(N+2) = 299
IPL(N+3) = 287
IPL(N+4) = 295
IPL(N+5) = 303
N = N+6
RETURN
END

C11A003C
C11A001C
C11A002C
C11A007C
C11A004C
C11A005C
C11A010C
C11A011C
C11A012C
C11A013C
C11A014C
C11A015C
C11A016C
C11A017C
C11A020C

[illegible]


```

AD22=A32*PBA-A12*RBA
AD21=A31*PBA-A11*RBA
AD13=A23*PBA-A33*QBA
AD12=A22*PBA-A32*QBA
AD33=A13*PBA-A23*PBA
AD32=A12*PBA-A22*PBA
AD31=A11*PBA-A21*PBA
AD23=A33*PBA-A13*RBA
RETURN
END

```

CC

```

SUBROUTINE D3I
RETURN
END

```

CC

```

SUBROUTINE D3
RETURN
END

```

CC

```

SUBROUTINE D4I
RETURN
END

```

CC

```

SUBROUTINE D4
RETURN
END

```

CC

```

SUBROUTINE D5I
RETURN
END

```

CC

```

SUBROUTINE D5
RETURN
END

```

CC

```

SUBROUTINE G1I

```

CC

GRAVITATIONAL AND CYCLIC ACCELERATION INITIALIZATION MODULE G1IC

```

COMMON ((3415)

```

CC


```

C      EQUIVALENCE (C(0507), F)
C      EQUIVALENCE (C(0507), F)
C      IF GZRC = C, FLAT-EARTH GRAVITATIONAL FIELD
C      IF (GZRC .EQ. 1.0) GO TO 1
C      COMPUTE FLAT-EARTH GRAVITATIONAL ACCELERATION
C      GCX = C
C      GCY = C
C      GCZ = C
C      F = -ZTF + HC
C      F = -VZTF
C      RETURN
C      1
C      IF GZRC = 1, SPHERICAL GRAVITATIONAL FIELD
C      R = SQRT(XTP*XTF + YTP*YTF + (ZTP-ZS)*(ZTP-ZS))
C      FD = (XTF*VXTF + YTF*VYTF + (ZTP-ZS)*VZTF)/R
C      F = R - RE
C      COMPUTE SPHERICAL GRAVITATIONAL ACCELERATION
C      GMR = -GM/(R*R*R)
C      GX = GMR*XTF
C      GY = GMR*YTF
C      GZ = GMR*(ZTP-ZS)
C      ADD CCRCLIS ACCELERATION
C      GCX = GX + CMGZ*VYTF - CMGY*VZTP
C      GCY = GY + CMGX*VZTP - CMGZ*VXTP
C      GCZ = GZ + CMGY*VXTP - CMGX*VYTP
C      RETURN
C      END
C      SUBROUTINE G2I
C      RETURN
C      END
C      SUBROUTINE G2
C      RETURN
C      END
C      SUBROUTINE G3I
C      COMMCN C(3415)
C      EQUIVALENCE (C(0208), TSA)
C      EQUIVALENCE (C(0257), CTS)
C      EQUIVALENCE (C(0258), STS)
C      EQUIVALENCE (C(0306), ZTP)
C      EQUIVALENCE (C(0414), FC)
C      EQUIVALENCE (C(0507), H)

```



```

EQUIVALENCE (C(0515), FSF)
EQUIVALENCE (C(0529), VAT)
EQUIVALENCE (C(0551), RICH)
EQUIVALENCE (C(0561), VAF)
F=FC-ZTF
ALT=F/3*280839895
VEL=VAT/3*280839895
CALL AIRF(ALT,VEL,-1,P,T,SON,DEN,Q,G,1,2)
CTS = CTS(TSA)
STS = SIN(TSA)
RETURN
END

```

CCC CCC

```

SUBROUTINE G3
AIR DATA MODULE G3
COMMON C(3415)
EQUIVALENCE (C(0224), A11)
EQUIVALENCE (C(0228), A12)
EQUIVALENCE (C(0232), A13)
EQUIVALENCE (C(0236), A21)
EQUIVALENCE (C(0240), A22)
EQUIVALENCE (C(0244), A23)
EQUIVALENCE (C(0248), A31)
EQUIVALENCE (C(0252), A32)
EQUIVALENCE (C(0256), A33)
EQUIVALENCE (C(0257), CTS)
EQUIVALENCE (C(0258), STS)
EQUIVALENCE (C(0282), AG)
EQUIVALENCE (C(0286), VXT)
EQUIVALENCE (C(0294), VYTP)
EQUIVALENCE (C(0302), VZTP)
EQUIVALENCE (C(0451), TWXTP)
EQUIVALENCE (C(0452), TWZTP)
EQUIVALENCE (C(0453), VATX)
EQUIVALENCE (C(0504), VATY)
EQUIVALENCE (C(0505), VATZ)
EQUIVALENCE (C(0506), F)
EQUIVALENCE (C(0507), GD)
EQUIVALENCE (C(0508), UBA)
EQUIVALENCE (C(0509), VBA)
EQUIVALENCE (C(0510), WBA)
EQUIVALENCE (C(0511), USA)
EQUIVALENCE (C(0512), VSA)
EQUIVALENCE (C(0513), )

```



```

A22=EE22
A23=EE23
A31=EE31
A32=EE32
A33=EE33
VELCCITY IN BODY AXES
UBA=A11*VATX+A12*VATY+A13*VATZ
VBA=A21*VATX+A22*VATY+A23*VATZ
WBA=A31*VATX+A32*VATY+A33*VATZ
VELCCITY IN STABILITY AXES
LSA=CTS*UBA+STS*WBA
VSA=VEA
WSA=-STS*UBA+CTS*WBA
ANGLE CF ATTACK AND SIDESLIP
ASA = ATAN(WSA/USA)
BSA = ARSIN(VSA/SQRT(LSA*USA + VSA*VSA + WSA*WSA))
RETURN
END

```

```

CC
SUBROUTINE G4I
RETURN
END

```

```

CC
SUBROUTINE G4
RETURN
END

```

```

CC
SUBROUTINE G5I

```

COORDINATE CCNVERSION INITIALIZATION MODULE

```

COMMON C(3415)
EQUIVALENCE (C(0557),CTPI11)
EQUIVALENCE (C(0558),CTPI12)
EQUIVALENCE (C(0559),CTPI13)
EQUIVALENCE (C(0560),CTPI21)
EQUIVALENCE (C(0562),CTPI22)
EQUIVALENCE (C(0563),CTPI23)
EQUIVALENCE (C(0564),CTPI31)
EQUIVALENCE (C(0565),CTPI32)
EQUIVALENCE (C(0566),CTPI33)
EQUIVALENCE (C(0576),ALAT)
EQUIVALENCE (C(0578),AZ)
AZ = AZ*.C1745329

```

T>Y0126C
 T>Y01270
 T>Y0128C


```

ELEV = ATAN (-ZTP/SCFT(XTP*XTP + YTP*YTP))
GO TC 6C
ELEV=0.C
60 CONTINUE
C CBI11 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI12 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI13 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI21 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI22 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI23 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI31 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI32 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI33 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C DETERMINE = ACCELERATION, VELOCITY AND POSITION IN ECI SYSTEM
AXECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
AYECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
AZECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
VXECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
VYECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
VZECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
XECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
YECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
ZECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
RETURN
END

C CBI11 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI12 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI13 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI21 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI22 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI23 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI31 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI32 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C CBI33 = CTPB11 + CTPB12 + CTPB13 + CTPB23
C DETERMINE = ACCELERATION, VELOCITY AND POSITION IN ECI SYSTEM
AXECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
AYECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
AZECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
VXECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
VYECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
VZECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
XECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
YECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
ZECI = CTPB11 + CTPB12 + CTPB13 + CTPB23
RETURN
END

SUBROUTINE G6I
RETURN
END

SUBROUTINE G6
RETURN
END

SUBROUTINE S1I
RETURN
END

Homing SEEKER INITIALIZATION MODULE S1I

COMMON C(3415)
DIMENSION IPL(100)
EQUIVALENCE (C(0317), FLGT)
EQUIVALENCE (C(0347), FLGTS)
EQUIVALENCE (C(3315), N)
EQUIVALENCE (C(3316), IPL(1))

```



```

IPL(N)=26366
IPL(N+1)=2270
IPL(N+2)=2314
IPL(N+3)=2226
IPL(N+4)=226
IPL(N+5)=416
IPL(N+6)=420
IPL(N+7)=424
IPL(N+8)=428
IPL(N+9)=428
N=N+10
FLGT=C:C
FLGTS=C:0
RETURN
END

```

CCC CCC

```

SUBROUTINE S1
  HCMING SEEKER MODULE S1
  CMNCA C(3415)
  EQUIVALENCE(C(187), CNCYS1)
  EQUIVALENCE(C(197), CNCIS2)
  EQUIVALENCE(C(212), P)
  EQUIVALENCE(C(216), Q)
  EQUIVALENCE(C(220), CMGYD)
  EQUIVALENCE(C(262), CMGY)
  EQUIVALENCE(C(265), CMGZ)
  EQUIVALENCE(C(266), CMGZ)
  EQUIVALENCE(C(269), THTS)
  EQUIVALENCE(C(270), THTS)
  EQUIVALENCE(C(273), PSIS)
  EQUIVALENCE(C(274), PSIS)
  EQUIVALENCE(C(277), CMGYS)
  EQUIVALENCE(C(279), CMGYS)
  EQUIVALENCE(C(280), AS)
  EQUIVALENCE(C(316), RS)
  EQUIVALENCE(C(317), FLGT)
  EQUIVALENCE(C(318), RMT)
  EQUIVALENCE(C(319), XLT)
  EQUIVALENCE(C(320), YDT)
  EQUIVALENCE(C(321), ZDT)
  EQUIVALENCE(C(322), CMZCID)
  EQUIVALENCE(C(325), CMZC1)
  EQUIVALENCE(C(326), CMYC1)
  EQUIVALENCE(C(329), CMYCI)

```



```

C      AZA=ATAN(YCA/RXZA)
      KEEP SEEKER (AGED IF POINTING ERRORS EXCEED GIMBAL LIMITS
      ELA1=ELA
      AZA1=AZA-4./RAD
      IF (ABS(ELA1)).GT.GMBLIM) GO TO 200
      IF (ABS(AZA1)).GT.GMBLIM) GO TO 200
      PRECESS SEEKER TO SEARCH INITIATION PT
      CMGYC=AKS*ELA1
      CMGZC=AKS*AZA1
      GO TO 50
C      SEARCH SEQUENCE
      10 IF (FLAGS.GT.0.0) GC TC 20
      TIME=T
      FLAGS=1.C
      WRITE (6,150) RMA,T
      20 IF (FLGTS.GT.0.0) GC TO 75
      TS = T-TIME
      IF (TS.GT.0.4) GO TC 30
      CMGYC=-10./RAD
      CMGZC=C
      CMGC=ABS(CMGYC)
      GO TC 50
      30 IF (TS.GT.1.35) GC TC 40
      CMGYC=C
      CMGZC=10./RAD
      CMGC=ABS(CMGZC)
      GO TC 50
      40 IF (TS.GT.1.775) GC TC 50
      CMGYC=TRLIM
      CMGZC=C
      CMGC=ABS(CMGYC)
      GO TC 50
      50 IF (TS.GT.2.4) GO TC 60
      CMGYC=C
      CMGZC=-TRLIM
      CMGC=ABS(CMGZC)
      GO TC 50
      60 IF (TS.GT.2.625) GO TC 65
      CMGYC=-TRLIM
      CMGZC=C
      CMGC=ABS(CMGYC)
      GO TC 50
      65 IF (TS.GT.3.425) GO TC 70
      CMGYC=C
      CMGZC=TRLIM
      CMGC=ABS(CMGZC)
      GO TC 50
      C      END CF SEARCH

```



```

70 IF (FLGCL.GT.0.0) GO TC 90
   CMGZC=C.C
   CMGYC=C.C
   WRITE (6,130)
   FLGTS=1.C
   GO TC 90
75 IF (FLGCL.GT.0.0) GC TC 80
   GO TC 90
80 REFCST=1.C
80 IF (FLGCL.GT.0.0) GC TC 85
   RYZT=SGRT(YDT*2+ZDT*2)
   CMGYC=-TRLIM*ZDT/RYZT
   CMGZC=TRLIM*YDT/RYZT
   IF (ANGLT.LE.0.25) FLGT=1.0
   GO TC 90
C   RADIO METER AND SENSOR FILTER LAGS
E5 WS=201.7
   WSF=125.7
   DIYD=WS*(.2*ERSEL-DIY)
   DIZD=WS*(.2*ERSAZ-DIZ)
   RECEIV=3.C
   SGMNQI=3.C
   ELNCIS=0.2
   AZNDCIS=C.2
   ADD NCISE TO FILTERED RADIO METER SIGNAL
   DIYN=DIZ+AZNDCIS
   CMYCIC=WSF*(AKS*DIYN/RAD-OMYC1)
   OMZCIC=WSF*(AKS*DIZN/RAD-OMZC1)
   CMGZC=OMZC1
   CMGYC=CMYC1
C   IF SEEKER LCSES TGT, PRECESSION COMMANDS GO TC ZERO
88 IF (ABS(ANGLT).LE.2.5) GC TO 90
   IF (FLAGLT.GT.0.0) GC TC 90
   WRITE (6,140) ANGLT
   FLAGLT=1.0
C   PRECESSION RATES
90 SEEKER FRECESSION RATES
   IF (ABS(CMGYC).GE.TRLIM) OMGYC=SIGN(TRLIM,CMGYC)
   IF (ABS(CMGZC).GE.TRLIM) OMGZC=SIGN(TRLIM,CMGZC)
   RESOLVE GIMBAL Y-AXIS ANGLE RATE TO SEEKER Y-AXIS
   OMGZC=CMGYC*CPSSIS+SPSSIS*(R*STHTS-P*CTHTS)
C   SEEKER GYMNAMICS
   D2YD=2.4*(CMGYC-OMGYC)
   CMGYC=2.129*(D2YD+D2YD/64.3)
   D2ZD=4.4*(CMGZC-OMGZC)
   CMGZC=4.4*(D2ZD+D2ZD/64.3)
C   SEEKER EULER ANGLE RATES
   THTS=CMGY-6

```



```

PSISD=CMGZ-P*STHTS-R*CTHTS
THILIM=SIGN(GMBLIM,THTS)
PSILIM=SIGN(GMBLIM,PSIS)
IF(AES(THTS).GT.GMBLIM) THTS=50.*(THILIM-THTS)
IF(ABS(PSIS).GT.GMBLIM) PSISD=50.*(PSILIM-PSIS)
RETURN
FORMAT (1H,10X,'END CF SEARCH,NO TARGET DETECTION')
FORMAT (1F,10X,'SEEKER LCST TARGET,LCCK ANGLE=',F8.3)
FORMAT (1F,10X,'START SEARCH,ATIGS TGT RGE=',F10.3,3X,'T=',F8.3)
END

```

CC C C

SUBRCUTINE S2I

```

COMMON C(3415)
EQUIVALENCE (C(0336), FLGD)
EQUIVALENCE (C(0347), FLGTS)
FLGD=C.C
FLGTS=C.C
RETURN
END

```

CC CC CC

SUBRCUTINE S2

RADICMETER MODULE S2

```

COMMON C(3415)
EQUIVALENCE (C(0172), PTAD2)
EQUIVALENCE (C(0224), A11)
EQUIVALENCE (C(0228), A12)
EQUIVALENCE (C(0232), A13)
EQUIVALENCE (C(0236), A21)
EQUIVALENCE (C(0240), A22)
EQUIVALENCE (C(0244), A23)
EQUIVALENCE (C(0248), A31)
EQUIVALENCE (C(0252), A32)
EQUIVALENCE (C(0256), A33)
EQUIVALENCE (C(0273), THTS)
EQUIVALENCE (C(0277), PSIS)
EQUIVALENCE (C(0280), ANGLT)
EQUIVALENCE (C(0290), XTP)
EQUIVALENCE (C(0298), YTF)
EQUIVALENCE (C(0306), ZTP)
EQUIVALENCE (C(0310), XTTP)
EQUIVALENCE (C(0311), YTTP)
EQUIVALENCE (C(0312), ZTTP)

```



```

STHTS=SIN(THTS)
CPSTIS=CCS(PSTIS)
SPSTIS=CSIN*CTHTS
B112=SPSTIS
B113=-CPSTIS
B211=SPSTIS
B212=-CPSTIS
B223=SPSTIS
B311=SPSTIS
B322=CTHTS
B333=CTHTS
RESOLVE1=XMT+B12*YMT+B13*ZMT
YDT=B211*XMT+B222*YMT+B233*ZMT
ZDT=B311*XMT+B322*YMT+B333*ZMT
RYZT=SCRT(XCT**2+YCT**2)
RXYT=SCRT(XCT**2+ZCT**2)
RESOLVE2=XMA+B12*YMA+B13*ZMA
YDA=B211*XMA+B222*YMA+B233*ZMA
ZDA=B311*XMA+B322*YMA+B333*ZMA
IF (FLAG5.GT.0.0) GC TC 5
FLG1=0.0
GO TC 20
5 ANGLT=ATAN(RYZT/XCT)
ANGLT=ANGLT*RAD
IF (FLGTS.GT.0.0) GC TO 30
BETA=6.27*0.7E-09*SCRT(CMGC*RAD/20.)*RMA**2
BTAC2=BETA/2.0
TEST FCR TARGET DETECTION
IF (FLG1.GT.0.0) GO TC 10
FLG1=SET WHEN TARGET ENTERS BEAM
IF (ABS(ANGLT).LE.BTAC2) FLG1=1.0
GO TC 20
10 IF (FLG1.GE.1.0) GC TC 20
TME=T
FLGD=SET WHEN TARGET LEAVES BEAM
IF (ABS(ANGLT).GE.BTAC2) FLGD=1.0
GO TC 20
20 TDE=T-TME
DETECTION DECLARED 30 MSEC AFTER TGT LEAVES BEAM
IF (TC.GE.C.03) GO TC 25
GO TC 200
25 IF (FLGTS.GE.1.0) GC TC 30
WRITE(6,160) T
FLGTS=1.0

```



```

C
30  GO TC 200
   IF (FLGT.LE.0.0) GO TC 200
   RADIO METER ERROR SIGNAL
   CHI1=2.0193*(2.*ANGLT/3.-1.0)
   CHI2=2.0193*(2.*ANGLT/3.+1.0)
   IF (CHI1.EC.0.0) GO TC 40
   ERSIG=(SIN(CHI1)/CHI1)**2-(SIN(CHI2)/CHI2)**2
   GO TC 50
40  ERSIG=1.0-(SIN(CHI2)/CHI2)**2
   ERSIG=ZEROC IF SFEKER LCSES TPACK
50  IF (ANGLT.GT.2.5) ERSIG=0.0
   RESCLVE INTC AZ,EL ERRCR SIGNALS (SMALL ANGLE APPROX)
   IF (RYZT.LE.0.0) GO TC 60
   ERSZ=ERSIG*YDT/RYZT
   ERSZ=ERSIG*(-ZDT)/RYZT
   GO TC 200
60  ERSZ=C.0
   ERSZ=C.0
200 RETURN
160 FORMAT (1F,10X,'END OF SEARCH,TARGET DETECTED.T=',F8.3)
      END
C
SUBROUTINE S2I
  RETURN
  END
C
SUBROUTINE S3
  RETURN
  END
C
SUBROUTINE S4I
  RETURN
  END
C
SUBROUTINE S4
  RETURN
  END
C
SUBROUTINE S5I
  RETURN
  END
C

```



```

C      AZA=EA31*FX1+BA32*AY1+BA33*AZ1
ACCELEROMETER QUANT. STEP=.194 G
AQS=6.24176
ANSX=AXA/ACS
AXMA=ACS*AIN(ANSX)
ANSY=AYA/ACS
AYMA=ACS*AIN(ANSY)
ANSZ=AZA/ACS
AZMA=ACS*AIN(ANSZ)
TRANSFORMATION BACK TC ECDY AXES
AXM=BA11*AXMA+BA21*AYMA+BA31*AZMA
AYM=BA12*AXMA+BA22*AYMA+BA32*AZMA
AZM=BA13*AXMA+BA23*AYMA+BA33*AZMA
*** ICEAL CYRCS *****
20 CM=G
   RM=R
   FM=P
   RETURN
   END
C      SUBROUTINE S6I
C      RETURN
C      END
C      SUBROUTINE S6
C      RETURN
C      END
C      SUBROUTINE S7I
C      RETURN
C      END
C      SUBROUTINE S7
C      RETURN
C      END
C      SUBROUTINE S8I
C      RETURN
C      END
C      SUBROUTINE S8
C      RETURN
C      END

```



```

C      4A(18)/401.847/,W(19)/404.344/,W(20)/406.206/,W(21)/407.368/,
      5W(22)/408.396/,W(23)/409.175/,W(24)/409.883/,W(25)/410.296/

      TIME=TI-TI
      IF(TIME.GT.C.) GO TC 3
      FF=0.0
      THRUST=C.O
      GO TC 4
      DO 1 I=2,25
      IF(T(I).GE.TIME) GC TC 2
      CONTINUE
      CALL EXIT
      2 AK=(TIME-T(I-1))/(T(I)-T(I-1))
      THRUST=F(I-1)+(F(I)-F(I-1))*AK
      FF=W(I-1)+(W(I)-W(I-1))*AK
      PSF=2116.217*(1.C-(6.87534703E-06)*H)**5.25627005)
      THRUST=THRUST+(2116.2-PSF)*.715428
      RETURN
      4 END

      SUERCUTINE RAMJET (FIN,XMIN,AIN,WFIN,XIN,CT)

      SIMPLIFIED RAMJET MODEL
      USES TABLE LOCK-UP FOR THRUST COEFF AND FUEL FLOW RATE

      DIMENSION TM1(6),TM2(7),TH(4),TETA(5),TCF1(6,5,4),TCF2(7,5,4),
      1TW(6,5,4)
      COMMON C(3415)

      DATA TM1/2.3,2.5,2.7,2.9,3.1,3.3/
      DATA TM2/C.5,1.0,1.5,2.0,2.5,3.0,3.3/
      DATA TH/500.,1300.,2000.,3000.,35000./
      DATA TETA/C.,2.,4.,6.,8./

      DATA TCF1/ .3814,.2958,.2075,.1374,
      A.0844,.0443,.3794,.2843,.2025,.1339,.0805,.0402,.3731,.2818,
      B.1915,.1203,.0666,.0258,.3476,.2526,.1695,.1049,.0555,.0173,
      C.3251,.2355,.1595,.1001,.0549,.0291,.4357,.3412,.2453,.1651,
      D.1115,.0682,.4337,.2293,.2401,.1654,.1075,.0639,.4273,.3268,
      E.2287,.1512,.0927,.0485,.4012,.2967,.2061,.1350,.0810,.0396,
      F.3782,.2751,.1954,.1300,.0805,.0425,.4944,.386,.2798,.1963,
      G.1341,.088,.4924,.373,.2742,.1924,.1256,.0832,.4857,.3706,.2618,
      H.1766,.1132,.066,.4573,.3376,.237,.1588,.1003,.056,.4328,.3186,

```


I-2254, .1535, .0999, .0555, .6273, .5139, .3561, .2997, .2243, .1661, .2054,
 J-625, .5028, .3911, .256, .2202, .1616, .6175, .4598, .3801, .2821, .2054,
 K-1459, .5531, .4721, .3588, .2662, .1936, .1367, .5708, .4555, .3484,
 L-2613, .1932, .1398/

C

DATA TCF2/-5727, -3661, -4238, 5727, -3658, -4235, -2618, -2067,
 A-2615, -2067, -2074, -2091, -4233, -2623, -2069, -2085,
 B-2072, -2069, -2074, -2091, -4233, -2623, -2069, -2085,
 C-5729, -2631, -2064, -2081, -3645, -4226, -2058,
 D-4205, -2060, -2075, -3646, -2053, -2073, -4194,
 E-2074, -2060, -2075, -3646, -2053, -2073, -4194,
 F-3636, -2054, -2075, -3646, -2053, -2073, -4194,
 G-2074, -2060, -2075, -3646, -2053, -2073, -4194,
 H-3636, -2054, -2075, -3646, -2053, -2073, -4194,
 I-2041, -2054, -2075, -3646, -2053, -2073, -4194,
 J-5666, -2054, -2075, -3646, -2053, -2073, -4194,
 K-4154, -2054, -2075, -3646, -2053, -2073, -4194,
 L-2043, -2054, -2075, -3646, -2053, -2073, -4194,
 M-2043, -2054, -2075, -3646, -2053, -2073, -4194,
 N-3556, -2054, -2075, -3646, -2053, -2073, -4194,
 O-2589, -2054, -2075, -3646, -2053, -2073, -4194,
 P-2017, -2054, -2075, -3646, -2053, -2073, -4194/

C

DATA TWF/ 2.95, 2.95, 2.885, 2.835, 2.771, 2.763, 2.911, 2.867,
 A-806, 2.806, 2.882, 2.853, 2.835, 2.819, 2.806, 2.778, 2.763, 2.748, 2.733, 2.718, 2.703, 2.688, 2.673,
 B-2.604, 2.571, 2.538, 2.505, 2.472, 2.439, 2.406, 2.373, 2.340, 2.307, 2.274, 2.241, 2.208, 2.175, 2.142, 2.109, 2.076, 2.043, 2.010, 1.977, 1.944, 1.911, 1.878, 1.845, 1.812, 1.779, 1.746, 1.713, 1.680, 1.647, 1.614, 1.581, 1.548, 1.515, 1.482, 1.449, 1.416, 1.383, 1.350, 1.317, 1.284, 1.251, 1.218, 1.185, 1.152, 1.119, 1.086, 1.053, 1.020, 0.987, 0.954, 0.921, 0.888, 0.855, 0.822, 0.789, 0.756, 0.723, 0.690, 0.657, 0.624, 0.591, 0.558, 0.525, 0.492, 0.459, 0.426, 0.393, 0.360, 0.327, 0.294, 0.261, 0.228, 0.195, 0.162, 0.129, 0.096, 0.063, 0.030, 0.000,
 C-1.876, 1.779, 1.712, 1.645, 1.578, 1.511, 1.444, 1.377, 1.310, 1.243, 1.176, 1.109, 1.042, 0.975, 0.908, 0.841, 0.774, 0.707, 0.640, 0.573, 0.506, 0.439, 0.372, 0.305, 0.238, 0.171, 0.104, 0.037, 0.000,
 D-1.812, 1.715, 1.648, 1.581, 1.514, 1.447, 1.380, 1.313, 1.246, 1.179, 1.112, 1.045, 0.978, 0.911, 0.844, 0.777, 0.710, 0.643, 0.576, 0.509, 0.442, 0.375, 0.308, 0.241, 0.174, 0.107, 0.040, 0.000,
 E-1.737, 1.640, 1.573, 1.506, 1.439, 1.372, 1.305, 1.238, 1.171, 1.104, 1.037, 0.970, 0.903, 0.836, 0.769, 0.702, 0.635, 0.568, 0.501, 0.434, 0.367, 0.300, 0.233, 0.166, 0.099, 0.032, 0.000,
 F-1.557, 1.460, 1.393, 1.326, 1.259, 1.192, 1.125, 1.058, 0.991, 0.924, 0.857, 0.790, 0.723, 0.656, 0.589, 0.522, 0.455, 0.388, 0.321, 0.254, 0.187, 0.120, 0.053, 0.000,
 G-1.421, 1.324, 1.257, 1.190, 1.123, 1.056, 0.989, 0.922, 0.855, 0.788, 0.721, 0.654, 0.587, 0.520, 0.453, 0.386, 0.319, 0.252, 0.185, 0.118, 0.051, 0.000,
 H-1.373, 1.276, 1.209, 1.142, 1.075, 1.008, 0.941, 0.874, 0.807, 0.740, 0.673, 0.606, 0.539, 0.472, 0.405, 0.338, 0.271, 0.204, 0.137, 0.070, 0.000,
 I-1.333, 1.236, 1.169, 1.102, 1.035, 0.968, 0.901, 0.834, 0.767, 0.700, 0.633, 0.566, 0.499, 0.432, 0.365, 0.298, 0.231, 0.164, 0.097, 0.030, 0.000,
 J-872, .867, .862, .857, .852, .847, .842, .837, .832, .827, .822, .817, .812, .807, .802, .797, .792, .787, .782, .777, .772, .767, .762, .757, .752, .747, .742, .737, .732, .727, .722, .717, .712, .707, .702, .697, .692, .687, .682, .677, .672, .667, .662, .657, .652, .647, .642, .637, .632, .627, .622, .617, .612, .607, .602, .597, .592, .587, .582, .577, .572, .567, .562, .557, .552, .547, .542, .537, .532, .527, .522, .517, .512, .507, .502, .497, .492, .487, .482, .477, .472, .467, .462, .457, .452, .447, .442, .437, .432, .427, .422, .417, .412, .407, .402, .397, .392, .387, .382, .377, .372, .367, .362, .357, .352, .347, .342, .337, .332, .327, .322, .317, .312, .307, .302, .297, .292, .287, .282, .277, .272, .267, .262, .257, .252, .247, .242, .237, .232, .227, .222, .217, .212, .207, .202, .197, .192, .187, .182, .177, .172, .167, .162, .157, .152, .147, .142, .137, .132, .127, .122, .117, .112, .107, .102, .097, .092, .087, .082, .077, .072, .067, .062, .057, .052, .047, .042, .037, .032, .027, .022, .017, .012, .007, .002, .000,
 K-872, .867, .862, .857, .852, .847, .842, .837, .832, .827, .822, .817, .812, .807, .802, .797, .792, .787, .782, .777, .772, .767, .762, .757, .752, .747, .742, .737, .732, .727, .722, .717, .712, .707, .702, .697, .692, .687, .682, .677, .672, .667, .662, .657, .652, .647, .642, .637, .632, .627, .622, .617, .612, .607, .602, .597, .592, .587, .582, .577, .572, .567, .562, .557, .552, .547, .542, .537, .532, .527, .522, .517, .512, .507, .502, .497, .492, .487, .482, .477, .472, .467, .462, .457, .452, .447, .442, .437, .432, .427, .422, .417, .412, .407, .402, .397, .392, .387, .382, .377, .372, .367, .362, .357, .352, .347, .342, .337, .332, .327, .322, .317, .312, .307, .302, .297, .292, .287, .282, .277, .272, .267, .262, .257, .252, .247, .242, .237, .232, .227, .222, .217, .212, .207, .202, .197, .192, .187, .182, .177, .172, .167, .162, .157, .152, .147, .142, .137, .132, .127, .122, .117, .112, .107, .102, .097, .092, .087, .082, .077, .072, .067, .062, .057, .052, .047, .042, .037, .032, .027, .022, .017, .012, .007, .002, .000/

C

F=FIN*3.280839895
 IF (XIN.GT.0.0) GO TC 10
 RCT=TH-RECL(XMIN,AIN,H,TM1,TETA,TH,TCF1,6,5,4)
 WFIN=0.0
 GO TC 20
 RCT=TH-RECL(XMIN,AIN,H,TM2,TETA,TH,TCF2,7,5,4)
 WFIN=0.0
 RETURN
 10
 20
 END

C

1 SUBROUTINE ENGINE (FIN,XMIN,AIN,WFIN,TIN,XIN,YIN,ZIN,CT,SMARG,
IFI,IFREAD)

NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
CPIA NCMENCLATURE

REAL ISF,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
REAL MACR,MACRS,MACF
INTEGER CCNT

COMMON/CRUS/ ENG(30),IENG(15)
COMMON/RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
COMMON/RJ/ ACR,BLEED
COMMON/RJ/ CDASUB,CDASUP,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
COMMON/RJ/ ER,ERRL,ERLL,ETAC,ETAN,F
COMMON/RJ/ FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
COMMON/RJ/ H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
COMMON/RJ/ CCUNT,ICCNV,ICCNVP,IFIRST
COMMON/RJ/ N,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
COMMON/RJ/ MACR,MACRS,PM
COMMON/RJ/ P,P1,P2,P3,P4,P5,P6,PT301,PRAR
COMMON/RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
COMMON/RJ/ PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PTIPRI,Q
COMMON/RJ/ RADEG,R,SA,SART,SQPTOT
COMMON/RJ/ T,T1,T2,T3,T4,T5,T6,TIME
COMMON/RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
COMMON/RJ/ V,V1,V2,V3,V4,V5,V6
COMMON/RJ/ WA,WFX,X,Y,Z

* * * * *

COMMON/EXFALS/ TABLE(1000)

COMMON/WRITCT/ WRIT(20)

NAMELIST /MCNT/ ENG,IENG

*** THE INFORMATION BELOW NOTES HOW THE NAMELIST
DATA IS INPUT AND WHAT THE INPUTS ARE ***

ENG(1)=A1/AR OR A1 (N SQ) SEE I6
ENG(2)=A2/AR OR A3 (N SQ) SEE I6
ENG(3)=A4/AR OR A4 (N SQ) SEE I6
ENG(4)=A5/AR OR A5 (N SQ) SEE I6


```

ENG(5)=A6/AR CR A6 (M SC) SEE I6
ENG(6)=AR (METRES SQUARED) COEFFICIENT
ENG(7)=CCB BURNER DRAG COEFFICIENT
ENG(8)=ETAC, NOZZLE THRU EFFICIENCY
ENG(9)=ETAC, NOZZLE THRU EFFICIENCY
ENG(10)=WF, FUEL FLOW RATE (KG/SEC) SEE I5
ENG(11)=ISP, STAR (FCRY GAS GEN SYSTEMS)
ENG(12)=PFI, CF PRIMARY GENERATOR SYSTEMS)
ENG(13)=BLEED FLOW PERCENTAGE/ 100.
ENG(14)=AIR FUEL STICICHEMETRIC (AFS)
ENG(15)=CAN, NOZZLE MASS FLOW COEFFICIENT

REAL MCCSE
DIMENSION MCCSE(176), EAGLE(176)

***
THE INFLTS BELOW ARE SWITCHES ***
IENG(1)=I1, ENGINE TYPE SWITCH
IF I1=0, RAMJET TYPE IF I1=1, GAS GEN TYPE
IENG(2)=I2, RAMJET TYPE EFF SWITCH
IF I2=0, ETAC IS SET BY INPUT
IF I2=1, ETAC IS SET IN A SUBROUTINE
IENG(3)=I3, STANDARD DAY
IF I3=2, I3, STANDARD DAY
IENG(4)=I4, REAL AIR SWITCH SOLUTION
IF I4=0, USE THE REAL AIR SWITCH SOLUTION
IF I4=1, USE THE GAMMA=1.4 SOLUTION
IENG(5)=I5, WFF OR SOLUTION SWITCH
IF I5=1, THE PROGRAM USES WFF IN THE SOLUTION
IF I5=2, THE PROGRAM USES WFF IN THE SOLUTION
IENG(6)=I6, INLET DATA TYPE SWITCH
IF I6=0, THE INLETS ARE RATIOS BASED CN AR
IF I6=1, THE AREAS ARE ABSOLUTE VALUES
IENG(7)=I7, THE PRINT EACH ITERATION SWITCH
IF I7=0, DO NOT PRINT DEBUG DATA EACH ITERATION
IF I7=1, DO NOT PRINT DEBUG DATA
IENG(8)=I8, COMBUSTION PRESSURE EFFECT SWITCH
IF I8=0, THERE IS NO PRESSURE EFFECT COMPUTED
IF I8=1, THE PRESSURE EFFECT ON COMBUSTION TEMP IS COMPUTED

EQUIVALENCE(TABLE(206),MCCSE(1))
EQUIVALENCE(TABLE(382),EAGLE(1))

*** EXAMPLE CF NAMELIST INPUT ***
$MCNT
ENG=.5256,.85,.85,.4444,.56,.1140092,0.,.98,.95,2.,.50,1.2,.04,13.8
IENG=C.0,2,C.1,0,0,0,
IEND

```



```

I3=IENG(3)
I4=IENG(4)
I5=IENG(5)
I6=IENG(6)
I7=IENG(7)
I8=IENG(8)
IF (I6.EC.0) A1=A1*AR
IF (I6.EC.0) A3=A3*AR
IF (I6.EC.C) A4=A4*AR
IF (I6.EC.0) A5X=A5*AR
IF (I6.EC.0) A6=A6*AR
IF (I6.EC.1) A5X=A5
A5=A5X*CNM
IF (IFIRST.EC.-1) GC TC 20
GO TC 20
20 CONTINUE
CC COMMENT INITIALIZE ,SET CONSTANTS AND WRITE AREAS
CC
CC AIRREATHING GEOMETRY ONLY CALLED ON FIRST PASS
CC CALCULATE GEOMETRY PARAMETERS
A4OA5=A4/A5
GAM=5.0/7.0
M4=FMARL(GAM,A4OA5)
PHI4=PFIM(GAM,M4)
A6CA5=A6/A5
M6=FMARH(GAM,A6OA5)
PHI6=PFIM(GAM,M6)
CC IDENTIFY AND PRINT PARAMETERS
CC
WRITE (6,420) A1
WRITE (6,440) A3
WRITE (6,450) A4
WRITE (6,460) A5X
WRITE (6,470) A6
WRITE (6,480) AR
WRITE (6,490) PHI4
WRITE (6,500) PHI6
WRITE (6,510) CDR
WRITE (6,520) CNM
20 CONTINUE
CC SET CONSTANTS AND INITIALIZE SUBROUTINES
CC G IN KG M/SEC SQ - N

```



```

C      RADEG=.572557779E+02
C      G=9.80665
C      START CALCULATIONS FOR EACH NEW POINT
C      COUNT=0
C      ICCNV=1
C      NEW ITERATIONS START HERE
C      40 ICCNV=C
C      CHECK TC SEE IF 100 ITERATIONS OF THE SAME PCINT HAVE BEEN MADE
C      IF (CCOUNT-100) 60,60,50
C      IF NCT CCNVERGED AFTER 100 ATTEMPTS, STOP
C      50 WRITE (6,530)
C      CALL EXIT
C      IF NCT CCNVERGED WITH LESS THAN 100 ATTEMPTS, TRY AGAIN
C      60 CONTINUE
C      IF (CCOUNT.GT.89) I7=1C
C      IF (CCOUNT.GT.89) I7=1C
C      IF (CCOUNT.NE.0) GO TO 50
C      IF (IFIRST.EC.-1) CALL AIR(H,M,-1,P,T,A,DEN,Q,G,1,1)
C      IF (IFIRST.EC.-1) GO TO 70
C      I3=I2+10C
C      CALL AIR(H,M,1,P,T,A,DEN,Q,G,1,1)
C      G=9.80665
C      IF (I4.EC.1) GO TO 8C
C      ENTHR=46.4278+.23945*(T-194.444)
C      V IN METRES PER SEC
C      V=M*A
C      ENRG IN CAL/GM
C      ENRG=(V**2.)/8368.
C      ENTHAL=ENTHR+ENRG
C      ECGM IN CAL/GM
C      ECGM=ENTHAL+ENRG
C      TT3 IN DEGREE RANKINE
C      TT3=539.11+(7.5164*ECGM)-(3.3559E-03*ECGM*ECGM)+(4.65976E-06*(ECGM
C      1**3.1))- (5.32956E-09*(ECGM**4.1))
C      TT3=TT3/1.8
C      GAMMA3=1.4048-(6.09654E-04*ECGM)+(2.12102E-06*ECGM*ECGM)-(5.56845E

```



```

1-09*(ECGM**2.)))+(6.45543E-12*(ECGM**4.))
GO TC 50
EC CONTINUE
GAMMA3=1.4
TT3=1/FT*TT(1.4,M)
TT3=TT3*FT*TT(1.4,M3)
9C CONTINUE
C
IF (IFIRST) 100,110,110
100 WRITE (6,540)
C
COMBLSTCR
C
IF (I2.EG.1) CALL ETA
CALL TAE (AFS)
V=-0.0
WA=10.0
N3=.001
MACRS=1.
CALL HCRA
C
CALCULATION
C
WRITE (6,230)
C
DIFFUSER
C
CALL HCF
GO TC 150
CONTINUE
110 IF (N.GT.1.07) GO TC 130
*** BEGIN SUBSONIC CALL SEQUENCE *****
IF (WF.LT..001) GO TC 120
PT=P/FFCPT(1.4,M)
PMOPT=FFNCPT(1.4,M)
TT=T/FT*TT(1.4,M)
ACR=.27661+2.4721*M-4.527*M*M+2.3036*M*M*M
AINF=A1*ACR*1.05
WA=(AINF*FT*PMOPT)/SGRT(TT)
CALL HCRA
120 CONTINUE
IF (WF.LT..001) ER=0.0
FAR=ER/AFS
CALL SLESEN (H,M,FAR,CF,WF,Q,AR)
ICNV=C
COUNT=C
GO TC 225
C ***** END OF SUBSONIC CALL SEQUENCE *****

```



```

130 CCNTINLE
C
C
C ***** WE SET SA HERE *****
IF (I5.EC.1) GO TO 140
IF (I5.EC.2) GO TO 150
WRITE (6,240)
C GC THIS ROUTE IF ER IS INPUT OR CHANGED IN HCRA
140 CCNTINLE
CALL FCFR
IF (IFUEL.EC.5) ER=0.0
FAR=ER/AFS
WF=FAR*WA
AFR=1./FAR
GO TC 16C
C GC THIS ROUTE IS WF IS INPUT OR SET IN HCRA
150 CCNTINLE
CALL FCFR
IF (IFUEL.EC.5) WF=0.0
FAR=WF/WA
ER=FAR*AFS
IF (FAR.EC.0.0) GO TC 1400
AFR=1.C/FAR
GO TC 16C
1400 AFR=0.0
C
C 160 CCNTINLE
TOTI=FICTI(GAMMA3,M3)
I3=TT3*TCIT3
SQRTCT=SQRT(TT3)
C
C TO FIND AIR SPECIFIC IMPULSE
C
C IF (I10.EC.0) GO TO 161
CCNTINLE
GO TC 16C
161 CCNTINLE
CALL INTR20 (TT3,ER,TT41,TABLE,1)
CALL INTR20 (TT3,ER,GAMMA4,TABLE,2)
CALL INTR20 (TT3,ER,MW4,TABLE,3)
C
C COMPUTE THE EFFECT OF PRESSURE ON TEMPERATURE
C
C IF (I8.EC.0) GO TO 162
G1=GAMMA4+1.C
G2=GAMMA4-1.0
G3=G1/G2

```



```

G4=2.*G1*((2./G1)**G3)
G5=SCRT(G4)
PT4=(SA*WA)/(G5*A5)
PT4ATN=F14/101360.
IF(PT4ATM.LT.2) PT4ATN=.2
IF(PT4ATN.GT.20.) PT4ATN=20.
IF(T13.LT.299.) GC TC 1615
IF(ER.GT.1.5) GO TO 1615
CTEMP=TC(T13,ER,PT4ATN)
CONTINUE
T14I=T14I+CTEMP
162 CCNTINLE
CC
CC
NCW FINISH CCMBUSTOR COMPUTATIONS
R=8214.34/MW4
WAX=WA/.4535924
THROAT=AEX/.C9290304
TTT=((T13*1.8)/1000.)*2.
BSV=(WA)*TTT)/THRCAI
IF(I2.EC.1) CALL ETA
TT4=ETAC*(TT4I-TT3)+TT3
BBB=2.C*R*(1.0+GAMMA4)*TT4/GAMMA4
BBB=AMAX1(400.,BBB)
SA=(1.0+FAR)*SQRT(BEE)
SART=SA/SQRTCT
165 CCNTINLE
CC
CC
COMMENT MAKE CALCULATIONS TAKING VARIABLE NCZZLE GAMMA INTO EFFECT
A4CA5=A4/A5
M4=FMARL(GAMMA4,A4QA5)
PF I4=PF IN(GAMMA4,M4)
CC
CC
***** WE COMPUTE CONDITIONS AT STATION 3 HERE *****
STRCG=23.9613
IF(I1.EQ.1) GO TO 17C
RAMJET COMPUTATIONS - FUEL ADDED IN THE INLETS
SARTA=STEFCG*SCRT((GAMMA3+1.)/GAMMA3)*(1.+FAR)
PF I3=(PF I4*SART/SARTA)+(GAMMA3*CDB*M3*M3)/(2.*XMCIR3*SARTA))
GO TC 18C
GAS GENERATOR SYSTEM COMPUTATIONS
SARTE=STEFCG*SCRT((GAMMA3+1.)/GAMMA3)
PHI3=(PF I4*SART/SARTE)-(PHIPRI*ISTAR*FAR)/(SARTB*SQPTOT))+((GAMMA
170 13*CCDB*M3*M3)/(2.*XMCIR3*SARTE))
180 CCNTINLE
M3=FMPHIN(GAMMA3,PHI3)

```



```

C      FMCFT3=FFMCFT(GAMMA3,M3)
C      PMOPT=FFMCP(1.4,M)
C      XMCIF3=FMCI(1.4,M3)
C
C      ***** WE NOW CALL THE INLET ROUTINE TO GET WA *****
C
C      CALL HCF
C
C      190 CONTINUE
C
C      ***** DEBUG OUTPUT *****
C
C      IF (I7.EC.C) GO TO 200
C
C      201 CONTINUE
C      FK=F/1000.
C      PK=P/1000.
C      F2K=F2/1000.
C      P3K=P3/1000.
C      P4K=P4/1000.
C      F5K=F5/1000.
C      P6K=P6/1000.
C      FTK=FT/1000.
C      FT1K=FT1/1000.
C      PT2K=PT2/1000.
C      PT3K=PT3/1000.
C      PT4K=PT4/1000.
C      PT5K=PT5/1000.
C      PT6K=PT6/1000.
C      GK=G/1000.
C      WRITE(6,250) A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
C      WRITE(6,260) ACR,BLEED
C      WRITE(6,270) CDASUB,CDASUP,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
C      WRITE(6,280) ER,ERRL,ERLL,ETAC,ETAN,F
C      WRITE(6,290) FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA
C      16 WRITE(6,300) H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
C      WRITE(6,310) COUNT,ICCNV,ICONVP,IFIRST
C      WRITE(6,320) M,M1,M2,M3,M4,M5,M6,MW,Mh1,Mh2,Mh3,Mh4,Mh5,Mh6
C      WRITE(6,330) MACP,MACPS,PM
C      WRITE(6,340) PK,PIK,P2K,P3K,P4K,P5K,P6K,PT3CI,PRAR
C      12 WRITE(6,350) PTK,PTIK,PT2K,PT3K,PT4K,PT5K,PT6K,PMOPT,PMOPT1,PMOPT
C      WRITE(6,360) PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,QK
C      WRITE(6,370) RADEG,F,SA,SART,SQRTGT
C      WRITE(6,380) T,T1,T2,T3,T4,T5,T6,TIME
C      WRITE(6,390) TT,TT1,TT2,TT3,TT4,TT5,TT6
C      WRITE(6,400) V,V1,V2,V3,V4,V5,V6
C      WRITE(6,410) WA,WF,X,Y,Z

```



```

C 200 CONTINUE
C  SET IFTS, INCREASE CCUNT AND CALCULATE PARAMETERS NOT IN COMMON
C  COUNT=CCUNT+1
C  CHECK CONVERGENCE, ICCNVF IS USED TO MAKE SOLUTION CONVERGE TWICE
C  IF CCNVERGED, PRINT. IF NOT, GO BACK AND TRY AGAIN
C
C  IF (IFIRST.EQ.-1) GO TO 225
C  IF (ICCNV+ICCNVP) 210,220,210
C  ICCNVF=ICCNV
C  GO TO 40
C  CONTINUE
C  ***** WE NOW COMPUTE THRUST IF CONVERGED *****
C
C  A6CA5=A6/A5
C  N6=FMARF(GAMMA4,A6CA5)
C  PHI6=PTIN(GAMMA4,M6)
C  IF (I10.GE.5) GO TO 221
C  CF6=(WA*SA*PTI6*ETAN-A6*F)/(AR*Q)
C  CONTINUE
C  CFINF=2.*C*ACR*MACRS*A1/AR
C  CALPHA=CCS(ALPHA/RADEG)
C  CF=CF6-(CFINF*CALPHA)+CFB+CFC
C  F=CF*C*AR
C  IF(WF.EQ.0.0) GO TO 1300
C  ISF=F/(WF*G)
C  GO TO 1500
C  ISF=C.C
C  CONTINUE
C  G1=GAMMA4+1.C
C  G2=GAMMA4-1.0
C  G3=G1/G2
C  G4=2.*G1*((2./G1)**GE)
C  G5=SGRT(G4)
C  PT5=(SA*WA)/(G5*A5)
C
C  PREPARE DATA FOR TRANSFER TO MAIN PROGRAM
C
C 225 CONTINUE
C  FIN=F
C  XMIN=M
C  AIN=ALPHA
C  WFIN=WF
C  CDADC=CCASUB+CDASUP+CFT
C  CT=CF-CCACC

```



```

SMARG=FM
IFI=IFIRST
FCCUNT=CCUNT+.01
WRIT(1)=PT3/CI
WRIT(2)=WA
WRIT(3)=TT4
WRIT(4)=MACR
WRIT(5)=ACR
WRIT(6)=ER
WRIT(7)=IS
WRIT(8)=FCCUNT
WRIT(9)=WF
WRIT(10)=ETAC
WRIT(11)=CT
WRIT(12)=FMCUNT
WRIT(13)=ESV
WRIT(14)=EG.C.C) GO TC 120C
IF(PT3.EG.C.C)=PT5/PT3
WRIT(15)=PT5/PT3
CONTINUE
120C WRIT(16)=PT5/1000.
FCCUNT=FCCUNT
RETURN

```

```

C
2300 FORMAT (1F1,F8.2,8F7.4,2F7.3,F7.2)
2400 FORMAT (1F1,F8.5)
2500 FORMAT (1CF8.5)
2600 FORMAT (5F8.5,F10.3)
2700 FORMAT (5F8.5)
2800 FORMAT (F5.2,2F8.3,1CI2)
2900 FORMAT (4I4)
3000 FORMAT (7F7.4,7F8.4)
3100 FORMAT (2F5.5,F7.3)
3200 FORMAT (7F10.2,2F7.5)
3300 FORMAT (7F10.2,4F7.5)
3400 FORMAT (7F8.5,F10.2)
3500 FORMAT (F5.5,2F10.5,2F8.5)
3600 FORMAT (8F10.3)
3700 FORMAT (7F10.3)
3800 FORMAT (7F10.3)
3900 FORMAT (2F8.3,3F10.5)
4000 FORMAT (54F1,AIR,BREATHING ENGINE GEOMETRY
4100 FORMAT (14F,A 1 = F9.7,9H SQ METRE)
4200 FORMAT (14F,A 2 = F5.7,9H SQ METRE)
4300 FORMAT (14F,A 3 = F9.7,9H SQ METRE)
4400 FORMAT (14F,A 4 = F5.7,9H SQ METRE)
4500 FORMAT (14F,A 5 = F5.7,9H SQ METRE)
4600

```



```

470 FCFMAT (14H      A 6 = F9.7,9H SQ METRE)
480 FCFMAT (14H      A R = F9.7,9H SQ METRE)
490 FCFMAT (28H      PFI4 = F7.5)
500 FCFMAT (28H      PFI6 = F7.5)
510 FCFMAT (28H      BURNER DRAG COEFF. = F6.2)
520 FCFMAT (28H      BUZZLE MASS COEFF. = F6.2)
530 FCFMAT (26H      NCZLN DID NOT CONVERGE AFTER 100 ATTEMPTS)
540 FCFMAT (47H      RAMJET PROPULSION
                     )
END

SUBROUTINE ETA

NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
CPIA NCMENCLATURE

REAL ISF,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
REAL MACR,MACRS
INTEGER CCUNT

COMMON /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
COMMON /RJ/ ACR,BL,FEL
COMMON /RJ/ CDASUB,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
COMMON /RJ/ ER,ERRRL,ERLL,ETAC,ETAN,F
COMMON /RJ/ FAR,G,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
COMMON /RJ/ F,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
COMMON /RJ/ CCUNT,ICCNV,ICONVP,IFIRST
COMMON /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
COMMON /RJ/ MACR,MACRS,PM
COMMON /RJ/ P,P1,P2,P3,P4,P5,P6,PT30I,PRAR
COMMON /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
COMMON /RJ/ PHI,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPI,Q
COMMON /RJ/ RADEG,R,SA,SART,SQRTOT
COMMON /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
COMMON /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
COMMON /RJ/ V,V1,V2,V3,V4,V5,V6
COMMON /RJ/ WA,WF,X,Y,Z

DATA A1,E11,C11,D11,E11,F11,G11
* /-14C7.68, .95416, -8.55033, 43.7266, 1468.91, 80484.,
* -277C56./

FAR2=FAR*FAR
FOVER=1./(1.+FAR)

```



```

TERM1=B11+C11*FAR+D11*FAF2
TERM2=E11+F11*FAR+G11*FAR2
TTR=1.E*TT
TT4I=A11+TTR*TERM1+FCVER*TERM2
STT4I=SGRT(TT4I)
ETAC=40.246*STT4I-0.2513*TT4I-1055.08
ETAC=ETAC/100.
IF(ETAC.LT. .0001) ETAC=0.0
RETURN
END

SUBROUTINE FCRA
DIMENSION PS(66), PSTAT1(11), FUELF(11)

NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
CPIA NCMENCLOSURE

REAL ISP, ISTAR, M, M1, M2, M3, M4, M5, M6, MW, MW1, MW2, MW3, MW4, MW5, MW6
REAL MACR, MACRS
INTEGER CCUNT

COMMON /RJ/ A, AINF, A1, A2, A3, A4, A5, A6, AR, AFR, AFS, ALPHA
COMMON /RJ/ ACR, BLEED
COMMON /RJ/ CDASUB, CDASUF, CDB, CF, CFINF, CF6, CFB, CFT, CFC, CNM
COMMON /RJ/ ER, ERFL, ERLL, ETAC, ETAN, F
COMMON /RJ/ FAR, G, GAMMA, GAMMA1, GAMMA2, GAMMA3, GAMMA4, GAMMA5, GAMMA6
COMMON /RJ/ F, ISP, ISTAR, I1, I2, I3, I4, I5, I6, I7, I8, I9, I10
COMMON /RJ/ CCUNT, ICCNV, ICCNVP, IFIRST
COMMON /RJ/ M, M1, M2, M3, M4, M5, M6, MW, MW1, MW2, MW3, MW4, MW5, MW6
COMMON /RJ/ MACR, MACRS, PM
COMMON /RJ/ P, P1, P2, P3, P4, P5, P6, PT30I, PRAR
COMMON /RJ/ PT, PT1, PT2, PT3, PT4, PT5, PT6, PMOPT, PMOPT1, PMOPT2, PMOPT3
COMMON /RJ/ PH1, PH12, PH13, PH14, PH15, PH16, PH1PRI, Q
COMMON /RJ/ RADEG, R, SA, SART, SQRTOI
COMMON /RJ/ T, TT1, T2, T3, T4, T5, T6, TIME
COMMON /RJ/ TT, TT1, TT2, TT3, TT4, TT5, TT6
COMMON /RJ/ V, V1, V2, V3, V4, V5, V6
COMMON /RJ/ WA, WFX, Y, Z

ANGLE CF ATTACK PS(1) I=3,14
MACH NUMBER PS(1) I=15,81

DATA PS/

```



```

A12.0,4,C,C,0,0,5,1,0,1,5,2,0,
A2.5,3,C,4,C,5,0,6,C,7,C,8,0,
A1.5,2,C,2,5,3,0,89727,88386,
A.58407,93504,85727,88386,
A.58069,93504,85689,88275,
A.57828,93504,85689,88275,
A.57586,93410,89463,88165,
A.57345,93295,89313,87833,
A.57056,93039,89087,87501,
A.56815,92808,88711,87058,
A.56320,92158,87733,85842,
A.55704,91044,86076,84183,
A.54068,89257,83781,81745,
A.54111,87331,81185,78210,
A.52905,85267,79190,75666/

C DATA FSTAT1/3.07,3.9,4.9,6.02,7.8,9.1,10.11,12.13.18/
C
C DATA FUELF/.94,1.130,1.375,1.628,1.859,
A 2.056,2.250,2.416,2.562,2.722,3.05/
C
C IF (IFIRST) 10,20,20
10 CONTINUE
WRITE (6,50)
RETURN
20 CONTINUE
30 IF (CCOUNT-1) 30,30,40
CCOUNT=CCOUNT-1
CALL INTR2C (ALPHA,M,PS1CP,PS,1)
PS1=PS1CP*(P/6894.757)
TFFPS=CDLI(PS1,PSSTAT1,FUELF,11)
TFF=TFFPS*.4535924
WF=TFF
40 RETURN

C 50 FORMAT (T6,'FUEL CCNTRCL FOR STV G ')
C
C END
C
C SUBROUTINE FCF
C
C DIMENSION PR(81), AA(81), CADD(12), XM(12), BWR(12)
C
C AWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
CPIA NOMENCLATURE

```



```

REAL ISP,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
REAL MACR,MACRS
INTEGER CCNT

CCMCMCN /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
CCMCMCN /RJ/ ACR,BLEEC
CCMCMCN /RJ/ CDASUB,CDASUP,CDB,CF,CFINF,CF6,CFB,CFT,CFG,CNM
CCMCMCN /RJ/ ER,ERPL,ETAC,ETAN,F
CCMCMCN /RJ/ FAR,G,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
CCMCMCN /RJ/ H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
CCMCMCN /RJ/ CCUNT,ICCNV,ICCNVP,IFIRST
CCMCMCN /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
CCMCMCN /RJ/ MACR,MACRS,PM
CCMCMCN /RJ/ P,P1,P2,P3,P4,P5,P6,PT301,PRAR
CCMCMCN /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
CCMCMCN /RJ/ PHI,PHI2,PHI3,PHI4,PHI5,PHI6,PHI7,PHI8,PHI9,PHI10
CCMCMCN /RJ/ RADEG,R,SA,SART,SQRTOT
CCMCMCN /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
CCMCMCN /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
CCMCMCN /RJ/ V,V1,V2,V3,V4,V5,V6
CCMCMCN /RJ/ WA,WF,X,Y,Z

PRESSURE RECOVERY INSTALLED ALVJ -- BETA=0.0 (PS(I) I=15,81)

DATA FR/
A5.C,7.C,C,0,C,9,1,0,1,5,2,2,
A2.43,2,7,3,1,4,0,0,C,1,0,2,0,
A4.0,6,C,8,0,10,0,
A .5500,5475,9450,5400,9250,8775,8320,
A .9500,5475,9450,5400,9250,8775,8320,
A .9500,5475,9450,5400,9250,8775,8320,
A .5400,5375,9350,5300,9100,8600,8150,
A .7675,7660,7640,7600,7430,7000,6575,
A .6580,6860,6840,6800,6350,5750,5275,
A .5400,5375,5350,5300,4925,4300,3800,
A .3800,3760,3725,3650,3400,2850,2400,
A .2000,1560,1925,1850,1600,1050,0800/

AIR CAPTURE INSTALLED ALVRJ DEC 1975 AA(I) I=19,81

DATA AA/
A5.C,7.C,C,0,C,9,1,0,1,5,2,2,
A2.43,2,7,3,1,4,0,0,C,1,0,2,0,
A4.C,6,C,8,0,10,0,
A .5465,5457,9452,5440,9400,9350,9270,
A .5465,5457,9452,5440,9400,9350,9270,

```



```

C
C
A .5825,.5819,.5812,.5800,.5700,.5650,.5050,
A .6725,.6712,.6700,.6675,.6520,.6400,.6275,
A .5010,.5000,.8990,.8970,.8725,.8550,.8350,
A .5425,.5400,.9375,.9325,.9200,.9075,.8950,
A .5450,.5425,.9410,.9375,.9300,.9225,.9125,
A .5450,.5437,.9425,.9400,.9370,.9325,.9250,
A .5450,.5437,.9425,.5400,.9370,.9325,.9250/
C
C
DATA XM/
A .5,.5,1.,1.4,1.6,2.2,2.43,2.48,2.5,2.7,3.0,3.25/
C
C
DATA CALC/
A .0213,.0212,.0132,.125,.0988,.0221,.0184,.0173,.0171
B .0123,.0145,.0176/
C
C
DATA BWR/
A .0339,.0339,.0208,.0237,.0249,.0334,.0287,.0265,.0259,
B .0213,.0171,.0150/
C
C
IF (IFIRST) 10,20,20
1C CONTINUE
PRAR=0.0
TCL=.0004
WRITE (6,100) TCL
RETURN
C
C
20 CONTINUE
C
C
SET PRESENT PR/AR EQUAL TC PREVIOUS PR/AR
FRAR=FFAF
C
C
CALCULATE NEW PR/AR
C
C
TT=T/FTCTT(1.4,M)
TTRATC=SCRT(TT3/TT)
PRAR=TTRATC*FMOPT*MACRS*A1/(PMOPT3*A3)
IF (IBC.EQ.0) PRAR=TTRATC*PMOPT*MACRS*A1*(1.0+FAI)/(PMOPT3*A3)
C
C
CHECK FOR CONVERGENCE BY COMPARING PR/AR TO PREVIOUS PR/AR
C
C
IF (ABS(FRAR-PRAR)-TCL) 40,40,30
3C ICGNV=1
4C CONTINUE
CALPHA=CCS(ALPHA/RALFEG)
C

```



```

C          CALCULATE ADDITIVE DRAG DUE TO SUPERSONIC SPILLAGE
C
90  CAED=CILLI(N,XM,CADD,12)
   IF (PRAR.LE.PACR) CD$UB=C.0
   CCASUB=CCSUP*AL/AR
   CCASUP=CAED
   CFB=C.C
C
C          CFT IS RAT DRAG
CFT=C.CC$2
PESI=2.14678-1.11439*M+.22989*M*M
FE=PESI*F
AINF=AL*BIDCB
AE=9.33*.00064516
TRM1=P*AE
TRM2=P*AE
WDCT=((FT*AINFB*PMOPT)/SQRTINF
XMCE=WDCT*SCRT(TT-222.)/ (PE*AE)
XME=FRCN(1.4,XMCE)
TRM3=PE*AE*(1.+1.4*XME*XME)
CFC=-((TRM1+TRM2-TRM3)/(C*AR))
RETURN
C
100  FORMAT (T6,'ALVRJ INLET DATA DEC 1975 ,TOL=',F7.6)
C
C
C          FUNCTION INTRP (AH,AL,BF,BL,FH,FM)
C
REAL INTRP
C2 = AL + (AH - AL)*FM
C1 = BL + (BF - BL)*FM
INTRP = C1 + (C2 - C1)*FT
RETURN
END
C
C          FUNCTION FATAS(GAM,XM)
C
CFATAS  FUNCTION SUBPRCGAM TO CALCULATE A OVER ASTAR
C          FROM GAMMA AND MACF NUMBER
C
Z=(GAM+1.C)/(2.0*(GAM-1.C))
A=2.C/(GAM+1.0)
B=(GAM-1.0)/2.0
C=1.C/XM

```

FATA0030
FATA001C
FATA002C
FATA004C
FATA005C
FATA006C
FATA007C

FATA008C
FATA009C
FATA010C
FATA011C

R 2
R 4
R 6
R 8
R 10
R 12
R 14
R 16
R 18
R 20-

S 2
S 4
S 6
S 8
S 10
S 12
S 14
S 16
S 18
S 20-

FAR0030
FAR001C
FAR002C
FAR004C
FAR005C
FAR006C
FAR007C
FAR008C
FAR009C
FAR010C
FAR011C

C=A*(1.C+(E**XN**XM))
FATAS=C*(C**Z)
RETURN
END

FUNCTION FIXAC (A)

10 IFIX=ABS(A)-1.0000005
IF (TFIX.LT.0.0) GO TC 1C
IF (A.LT.0.0) GO TC 20
FIXAC=C.C
GO TC 3C
20 FIXAC=14159265
3C CONTINUE
RETURN
END

FUNCTION FIXAS (A)

10 IFIX=ABS(A)-1.0000005
IF (TFIX.LT.0.0) GO TC 1C
IF (A.LT.0.0) GO TC 20
FIXAS=1.57079633
GO TC 3C
20 FIXAS=-1.57079633
3C CONTINUE
RETURN
END

FUNCTION FMARH (GAM,ATAS)

CFMARH FUNCTION SUBPROGRAM TO CALCULATE MACH NUMBER
FCM ATAS AND GAMMA

500 Z = (GAM + 1.0) / (2.C*(GAM - 1.0))
DXMG = 10.0
XMG = 10.C
XMG = [XMG / 2.0
XMG = AMAX1(1.0,AMIN1(100.0,XMG))
IF (DXMG - .000015) 1CC,1C0,200
200 CATAS = 1.0/XMG*((2.0+(GAM - 1.0) *XMG **2)/(GAM + 1.0))**Z
IF (CATAS - ATAS) 700,100,800


```

700 XMG = XMG + DXMG
GO TO 500
800 XMG = XMG - DXMG
GO TO 500
100 FMARL = XMG
RETURN
END

```

CC C C C C C

FUNCTION FMARL(GAM,AR)

FUNCTION ROUTINE TO FIND MACH FROM A/A* AND GAMMA
USING NEWTONS METHOD FOR FINDING ROOTS OF AN EQUATION

```

SUBSCNIC SCLUTION
IF(GAM.LT.1.) WRITE(6,30)
IF(AR.LT.1.) WRITE(6,40)
IF(GAM.LT.1. .OR. AR.LT.1.) CALL EXIT
GM1=GAM-1.
GPI=GAM+1.
A=2./GPI
B=GM1/GPI
Z=GPI/(2.*GM1)
IF(AR.LT.1.3401) XMP=2.4706-(1.47059*AR)
IF(AR.GT.1.34 .AND. AR.LT.3.001) XMP=.77443-(.2048*AR)
IF(AF.GT.3.) XMP=(A*Z)/AR
TOL=.0001
CCNT=1
10 CONTINUE
BRACK=A+(B*XMP*XMP)
POWER=ERACK**Z
TOF=(1./XMP)-(AR/POWER)
BOTTCM=(1./BRACK)-(1./(XMP*XMP))
XM=XMP-(TCP/BOTTOM)
DELTA=ABS(XM-XMP)
XMP=XM
IF(DELTA.GT.TOL) GO TO 10

```

FMARL=XM

RETURN

```

FORMAT('GAMMA LT 1 IN FMARL')
FORMAT('A/A* LT 1 IN FMARL')
END

```

C C C C C C C

FMAR0120
FMAR0130
FMAR0140
FMAR0150
FMAR0160
FMAR0170
FMAR0180


```

C      FUNCTION FMPHIM (GAM , PHI )
C      CFMPHIM      FUNCTION SUBPRCGRAM TO CALC MACH NO      FROM PHIM
C
C      TOL = .CC001
C      XMSN = 1.0 / ( PHI**2 )
C      127 X = XMSN
C      XMSN = ( 1.0 + (GAM*(XMSN**2))) / (PHI * SQRT(2.0 * (GAM+1.0))*(1.0
C      1+((GAM-1.0)*.5*(XMSN**2))))
C      XMSN = AMAX1(0., AMIN1(1.0,XMSN))
C      IF (ABS(X-XMSN)-TOL) 125, 125, 128
C      128 XMSN = (X + XMSN) / 2.0
C      GO TO 127
C      125 FMPHIM = XMSN
C      RETURN
C      END
C
C      FUNCTION FMPOPT (GAM,POPT)
C      CFMPOPT      FUNCTION SUBPRCGRAM TO CALC XMACH
C      FROM P/PT AND GAMMA
C
C      Z = (FCPT) ** ((1.0 - GAM)/ GAM)
C      SMPPT = ( Z - 1.0)/(GAM - 1.0) * 2.0
C      FMPOPT = SQRT(SMPPT)
C      RETURN
C      END
C
C      FUNCTION PHIM (GAM, XMCH)
C      CFPHIM      TO CALC PHI FROM GAMMA AND MACH
C
C      PHIM = (1.0+GAM *(XMCH **2)) / (XMCH * SQRT(2.0*(GAM + 1.0))*(1.0+
C      1.5*(GAM-1.0) * (XMCH **2)))
C      RETURN
C      END
C
C      FUNCTION FPOPT (GAM,XMACH)
C      CFPCPT      FUNCTION SUBPRCGRAM TO CALC TOTAL PRESSURE RATIO
C      FROM GAMMA AND MACH NO.

```



```

C      Z = -GAM / (GAM - 1.0)
C      FPOPT = (1.0+.5*(GAM - 1.0)*(XMACH**2))**Z
C      RETURN
C      ENC
C
C      FUNCTION FRCM(GAM,XMCIR)
C
C      FUNCTION ROUTINE TO COMPUTE MACH NUMBERS FROM
C      MCIRCLE AND GAMMA FOR ALL MACH NUMBERS USING NEWTONS METHOD
C
C      IF (GAM.LT.1.) WRITE (6,100)
C      IF (GAM.LT.1.) CALL EXIT
C      A=GAM
C      B=0.5*(GAM*(GAM-1.))
C      C=XMCIR/.C5902063
C      XMP=C/(1.18+(.0923*C))
C      TCL=.0001
C      1C CONTINUE
C      BRACK=A+(B*XMP*XMP)
C      POWER=SQRT(BRACK)
C      TOP=(XMP*FCWER)-C
C      BOTTCM=(A+(2.*B*XMP*XMP))/POWER
C      XM=XMP-(TOP/BOTTCM)
C      DELTA=ABS(XM-XMP)
C      XMF=XM
C
C      IF (DELTA.GT.TOL) GO TO 1C
C
C      FRCM=XM
C
C      RETURN
C
C      100 FORMAT('GAMMA LT 1 IN FROM')
C      END
C
C
C      FUNCTION FTCT (GAM,XMACH)
C
C      FUNCTION SUBPROGRAM TO CALC T/TT FROM GAMMA AND MACH NC
C
C      FTCT = (1.0+.5*( GAM-1.0)*(XMACH**2))**(-1.0)
C      RETURN
C      END
C
C
C

```

FP000040
FP000050
FP000060
FP000070

FT000020
FT000030
FT000040
FT000050


```

DATA A/ 2210, -4.013517C, -2.2874890, -2.1765372,
A -4.40382210, -1.3585554, .0355105, .0498002,
B -0.0052258, .0061961, -.030910, -.0121108,
C D -0.026116, .0029277, -.0004221, .0044266,
E F -0.0010261, -.0039804, .0007523, .0013126,
G -0.0000001, .0033572, -.0005513, -.001147,
H -0.0001420, -.0021286, .0002918, .0008523,
      .0010109, .99998983, 1.0000000/

C ** SET CCNstants
C
C DATA PC,CC,RBAR,GC,Z1/
C * 101325.0,1.225,287.07299,9.80665,30480.0/
C
C ** IF NECESSARY , CONVERT TO SI UNITS
C      Z BECOMES METERS
C      V BECOMES METERS/SEC
C
C IF (K.EC.2) Z=Z*.3048
C IF (K.EC.2.AND.KK.EC.2) V=V*.3048
C
C ** START HERE
C
C IF (J) 10,20,20
C WRITE (6,6C)
C WRITE (6,7C) Z1
C
C ** CCMPUTE C(K) COEFFICIENTS
C
C 2C ETA=2.*(2.*Z/Z1-1.)
C C(1)=ETA
C C(2)=ETA**2-2.
C 3C I=3,15
C C(I)=ETA*C(I-1)-C(I-2)
C
C ** CCMPUTE CHEBYSHEV TRUNCATED EXPANSION
C
C ALNP=A(1,1)
C ALND=A(2,1)
C 4C I=2,15
C ALNF=ALNP+A(1,I)*C(I-1)
C ALND=ALND+A(2,I)*C(I-1)
C
C ** CCMPUTE ATMOSPHERIC PRESSURE
C      P IN N/SQ METER
C
C ALNP=.5*ALNP

```



```

**      POFZ=EXF(ALNP)
**      F = FCFZ*FC*A(1,16)
**      COMPUTE GRAVITY AT ALTITUDE AND LATITUDE
**      RC IS EQUATORIAL RADIUS (M) PER CRC HANDBOOK
**      GC IN M/SEC SQ
C C C C C

RC=6378377.45
GO=GC*((RC/(RO+Z))**2)
C C C C C

**      COMPUTE ATMOSPHERIC DENSITY
**      D IN KG PER CUBIC METRE
C C C C C

ALND=.5*ALND
CODZ=EXF(ALND)
L = CODZ*CC*A(2,16)
C C C C C

**      COMPUTE AIR TEMPERATURE
**      T IN DEGREE KELVIN
C C C C C

T=P/(REAR*D)
C C C C C

**      COMPUTE SPEED OF SOUND IN M/SEC
C C C C C

S=SQRT(401.90219*T)
C C C C C

**      COMPUTE DYNAMIC PRESSURE IN PA
C C C C C

IF (KK.EC.1) Q=0.7*F*V**2
IF (KK.EC.2) Q=0.7*F*(V/S)**2
C C C C C

**      CONVERT TO ENGLISH UNITS IF NECESSARY
C C C C C

IF (K.EC.1) GC TO 50
Z=Z/.3048
IF (KK.EC.2) V=V/.3048
P=P/6894.757
T=T*1.8
S=S/.3048
D=D/.16.C1846
C=C/6894.757
GC=GC/.3048
RETURN
50  FORMAT (T9,'*** US 1962 STANDARD ATMOSPHERE')
60  FORMAT (T9,'*** ALTITUDE FROM 0 TO ',F8.0,' METERS')
7C  END
C C C C C

```


C
C
C
C

```

FUNCTION ANORM(A,B)
CALCULATES TOTAL PRESSURE RATIO ACROSS A NORMAL SHOCK
A=GAMMA,E=XNACH
C=(A+1.)/2.
L=(A-1.)/2.
E=(2.*A)/(A+1.)
F=(A-1.)/(A+1.)
G=A/(A-1.)
H=1./(A-1.)
BB=B*B
ANUM=((C*EE)/(1.+C*EE))*G
CEM=((E*EE-F)**H)
ANCRM = ANUM/DEM
RETURN
END

```

C
C
C
C

```

FUNCTION FMCPT(GAM,XM)
FUNCTION SUBPROGRAM TO CALCULATE P/PT * M CIRCLE
FROM GAMMA AND MACH NUMBER
DIMENSION S = SEC * SQRT(K) / METRE
FPMOPT = FMCIR ( GAM , XM ) * FPOPT ( GAM , XM )
RETURN
END

```

C
C
C
C

```

SUBROUTINE SERCH (X,A,J,I,M)
DIMENSION A(20)
DO 10 L=1,I
J=L
IF (X.LE.A(L)) GO TO 20
CONTINUE
20 IF (J.EC.1) GO TO 30
J=J-1
RETURN
30 END

```

C
C
C
C

```

FUNCTION SHOCK(XM,D)

```

B	2
B	4
B	6
B	8
B	10
B	12
B	14
B	16
B	18
B	20-

30
32
34
36
38
40
42
44
46-
DDDDDDDD

2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
40-
EEEEEEEEEEEEEEEEEEEE

```

YY=TEY(J)
TXY=TAE(I,J)+(X-TBX(I))*((TAB(I+1,J)-TAB(I,J))/(TBX(I+1)-TBX(I)))
XJ=TAB(I,J)+(X-TBX(I))*((TAB(I+1,J)-TAB(I,J))/(TBX(I+1)-TBX(I)))
XJ1=TAE(I,J+1)+(X-TBX(I))*((TAB(I+1,J)-TAB(I,J))/(TBX(I+1)-TBX(I)))
1(I))
STDL=XJ+(Y-TBY(J))*((XJ1-XJ)/(TBY(J+1)-TBY(J)))
STDL1=STDL
RETURN
1C END

```

CC CCCCCCCCCC

FUNCTION STDLIA (TBX,TBY,TAB,X,Y,NX,NY)
 FUNCTION DESIGNED TO ALLOW THE USER TO INPUT X
 VARIABLE LENGTH TABLES. IT INTERPOLATES FIRST IN THE X
 VARIABLE AND THEN IN THE Y VARIABLE. IT EXTRAPOLATES FROM
 THE FIRST OR LAST INTERVAL IF THE X OR Y IS OUTSIDE OF THE
 BOUNDS OF THE TABLE. IT REQUIRES THE SUBROUTINE SERCH FOR
 ITS OPERATION.

PROGRAMMER -- F.SOBEL 30 OCTOBER 1967

```

DIMENSION TBX(NX), TBY(NY), TAB(NX,NY)
CALL SERCH (X,TBX,I,NX,IC)
CALL SERCH (Y,TBY,J,NY,JC)
XJ=TAB(I,J)+(X-TBX(I))*((TAB(I+1,J)-TAB(I,J))/(TBX(I+1)-TBX(I)))
XJ1=TAE(I,J+1)+(X-TBX(I))*((TAB(I+1,J)-TAB(I,J))/(TBX(I+1)-TBX(I)))
1(I))=X+(Y-TBY(J))*((XJ1-XJ)/(TBY(J+1)-TBY(J)))
STDLIA=STDLI
RETURN
10 END

```

CC C

SUBROUTINE SUBSON(H,ZM,F,C,W,Q,A)

```

C=.3
SFC=.3C
THR=.2248C9*C*Q*A
WF=(SFC*THR)/3600.
W=WF*.45359
IF(W.LT. 0.0)W=-W
RETURN
END

```

CC


```

C
F1=-((.227276E-3+.100807E-6*T)*T+.018785)
F2=((C(12)*C(97)-.1254E-5*T)*T+.0846043
F3=((C(11)*C(312E-7-.244175E-11*T)*T-.141615E-4)*T+.00496295)*T+.0056
A0473
C
C
C LINEARLY INTERPOLATE FCR FACTORS G1,G2,G3
DO 10 I=1,3
IF (ER.LE.E(I)) GO TO 20
CONTINUE
20 J=I-1
R=(ER-E(J))/(E(I)-E(J))
G1=G(J,1)+(G(I,1)-G(J,1))*R
G2=G(J,2)+(G(I,2)-G(J,2))*R
G3=G(J,3)+(G(I,3)-G(J,3))*R
C
C LINEARLY INTERPOLATE FCR FACTORS H1,H2
DO 30 I=1,13
IF (ATM.LE.P(I)) GO TO 40
CONTINUE
40 J=I-1
R=(ATM-P(J))/(P(I)-P(J))
H1=H(J,1)+(H(I,1)-H(J,1))*R
H2=H(J,2)+(H(I,2)-H(J,2))*R
C
C COMPUTE TEMPERATURE CORRECTION (TC)
TC=C(1)*F1*G1*H1+C(2)*F2*G2*H1+C(3)*F1*G2*H2+C(4)*F2*G1*H2
D
E
F
G
C
RETURN (6,110)
50 WRITE (6,110)
GO TO 100
60 WRITE (6,120)
GO TO 100
70 WRITE (6,130)
GO TO 100
80 WRITE (6,150)
GO TO 100
90 WRITE (6,140)
100 CALL EXIT
RETURN
C

```



```

110 FORMAT ('.0 TCTAL TEMPERATURE LESS THAN 298 K. END RUN.')
```

```

1120 FORMAT ('.0 TOTAL TEMPERATURE GREATER THAN 1300 K. END RUN.')
```

```

1130 FORMAT ('.0 FUEL EQUIVALENCE RATIO EXCEEDS 1.5 END OF RUN.')
```

```

1140 FORMAT ('.0 ATMOSPHERIC PRESSURE LESS THAN 0.2, END OF RUN.')
```

```

1150 FORMAT ('.0 ATMOSPHERIC PRESSURE EXCEEDS 20, END OF RUN.')
```

```

1160 END
```

```

1170
```

```

1180
```

```

1190
```

```

1200
```

```

1210
```

```

1220
```

```

1230
```

```

1240
```

```

1250
```

```

1260
```

```

1270
```

```

1280
```

```

1290
```

```

1300
```

```

1310
```

```

1320
```

```

1330
```

```

1340
```

```

1350
```

```

1360
```

```

1370
```

```

1380
```

```

1390
```

```

1400
```

```

1410
```

```

1420
```

```

1430
```

```

1440
```

```

1450
```

```

1460
```

```

1470
```

```

1480
```

```

1490
```

```

1500
```

```

1510
```

```

1520
```

```

1530
```

```

1540
```

```

1550
```

```

1560
```

```

1570
```

```

1580
```

```

1590
```

```

1600
```

```

1610
```

```

1620
```

```

1630
```

```

1640
```

```

1650
```

```

1660
```

```

1670
```

```

1680
```

```

1690
```

```

1700
```

```

1710
```

```

1720
```

```

1730
```

```

1740
```

```

1750
```

```

1760
```

```

1770
```

```

1780
```

```

1790
```

```

1800
```

```

1810
```

```

1820
```

```

1830
```

```

1840
```

```

1850
```

```

1860
```

```

1870
```

```

1880
```

```

1890
```

```

1900
```

```

1910
```

```

1920
```

```

1930
```

```

1940
```

```

1950
```

```

1960
```

```

1970
```

```

1980
```

```

1990
```

```

2000
```

```

2010
```

```

2020
```

```

2030
```

```

2040
```

```

2050
```

```

2060
```

```

2070
```

```

2080
```

```

2090
```

```

2100
```

```

2110
```

```

2120
```

```

2130
```

```

2140
```

```

2150
```

```

2160
```

```

2170
```

```

2180
```

```

2190
```

```

2200
```

```

2210
```

```

2220
```

```

2230
```

```

2240
```

```

2250
```

```

2260
```

```

2270
```

```

2280
```

```

2290
```

```

2300
```

```

2310
```

```

2320
```

```

2330
```

```

2340
```

```

2350
```

```

2360
```

```

2370
```

```

2380
```

```

2390
```

```

2400
```

```

2410
```

```

2420
```

```

2430
```

```

2440
```

```

2450
```

```

2460
```

```

2470
```

```

2480
```

```

2490
```

```

2500
```

```

2510
```

```

2520
```

```

2530
```

```

2540
```

```

2550
```

```

2560
```

```

2570
```

```

2580
```

```

2590
```

```

2600
```

```

2610
```

```

2620
```

```

2630
```

```

2640
```

```

2650
```

```

2660
```

```

2670
```

```

2680
```

```

2690
```

```

2700
```

```

2710
```

```

2720
```

```

2730
```

```

2740
```

```

2750
```

```

2760
```

```

2770
```

```

2780
```

```

2790
```

```

2800
```

```

2810
```

```

2820
```

```

2830
```

```

2840
```

```

2850
```

```

2860
```

```

2870
```

```

2880
```

```

2890
```

```

2900
```

```

2910
```

```

2920
```

```

2930
```

```

2940
```

```

2950
```

```

2960
```

```

2970
```

```

2980
```

```

2990
```

```

3000
```

```

3010
```

```

3020
```

```

3030
```

```

3040
```

```

3050
```

```

3060
```

```

3070
```

```

3080
```

```

3090
```

```

3100
```

```

3110
```

```

3120
```

```

3130
```

```

3140
```

```

3150
```

```

3160
```

```

3170
```

```

3180
```

```

3190
```

```

3200
```

```

3210
```

```

3220
```

```

3230
```

```

3240
```

```

3250
```

```

3260
```

```

3270
```

```

3280
```

```

3290
```

```

3300
```

```

3310
```

```

3320
```

```

3330
```

```

3340
```

```

3350
```

```

3360
```

```

3370
```

```

3380
```

```

3390
```

```

3400
```

```

3410
```

```

3420
```

```

3430
```

```

3440
```

```

3450
```

```

3460
```

```

3470
```

```

3480
```

```

3490
```

```

3500
```

```

3510
```

```

3520
```

```

3530
```

```

3540
```

```

3550
```

```

3560
```

```

3570
```

```

3580
```

```

3590
```

```

3600
```

```

3610
```

```

3620
```

```

3630
```

```

3640
```

```

3650
```

```

3660
```

```

3670
```

```

3680
```

```

3690
```

```

3700
```

```

3710
```

```

3720
```

```

3730
```

```

3740
```

```

3750
```

```

3760
```

```

3770
```

```

3780
```

```

3790
```

```

3800
```

```

3810
```

```

3820
```

```

3830
```

```

3840
```

```

3850
```

```

3860
```

```

3870
```

```

3880
```

```

3890
```

```

3900
```

```

3910
```

```

3920
```

```

3930
```

```

3940
```

```

3950
```

```

3960
```

```

3970
```

```

3980
```

```

3990
```

```

4000
```

```

4010
```

```

4020
```

```

4030
```

```

4040
```

```

4050
```

```

4060
```

```

4070
```

```

4080
```

```

4090
```

```

4100
```

```

4110
```

```

4120
```

```

4130
```

```

4140
```

```

4150
```

```

4160
```

```

4170
```

```

4180
```

```

4190
```

```

4200
```

```

4210
```

```

4220
```

```

4230
```

```

4240
```

```

4250
```

```

4260
```

```

4270
```

```

4280
```

```

4290
```

```

4300
```

```

4310
```

```

4320
```

```

4330
```

```

4340
```

```

4350
```

```

4360
```

```

4370
```

```

4380
```

```

4390
```

```

4400
```

```

4410
```

```

4420
```

```

4430
```

```

4440
```

```

4450
```

```

4460
```

```

4470
```

```

4480
```

```

4490
```

```

4500
```

```

4510
```

```

4520
```

```

4530
```

```

4540
```

```

4550
```

```

4560
```

```

4570
```

```

4580
```

```

4590
```

```

4600
```

```

4610
```

```

4620
```

```

4630
```

```

4640
```

```

4650
```

```

4660
```

```

4670
```

```

4680
```

```

4690
```

```

4700
```

```

4710
```

```

4720
```

```

4730
```

```

4740
```

```

4750
```

```

4760
```

```

4770
```

```

4780
```

```

4790
```

```

4800
```

```

4810
```

```

4820
```

```

4830
```

```

4840
```

```

4850
```

```

4860
```

```

4870
```

```

4880
```

```

4890
```

```

4900
```

```

4910
```

```

4920
```

```

4930
```

```

4940
```

```

4950
```

```

4960
```

```

4970
```

```

4980
```

```

4990
```

```

5000
```

```

5010
```

```

5020
```

```

5030
```

```

5040
```

```

5050
```

```

5060
```

```

5070
```

```

5080
```

```

5090
```

```

5100
```

```

5110
```

```

5120
```

```

5130
```

```

5140
```

```

5150
```

```

5160
```

```

5170
```

```

5180
```

```

5190
```

```

5200
```

```

5210
```

```

5220
```

```

5230
```

```

5240
```

```

5250
```

```

5260
```

```

5270
```

```

5280
```

```

5290
```

```

5300
```

```

5310
```

```

5320
```

```

5330
```

```

5340
```

```

5350
```

```

5360
```

```

5370
```

```

5380
```

```

5390
```

```

5400
```

```

5410
```

```

5420
```

```

5430
```

```

5440
```

```

5450
```

```

5460
```

```

5470
```

```

5480
```

```

5490
```

```

5500
```

```

5510
```

```

5520
```

```

5530
```

```

5540
```

```

5550
```

```

5560
```

```

5570
```

```

5580
```

```

5590
```

```

5600
```

```

5610
```

```

5620
```

```

5630
```

```

5640
```

```

5650
```

```

5660
```

```

5670
```

```

5680
```

```

5690
```

```

5700
```

```

5710
```

```

5720
```

```

5730
```

```

5740
```

```

5750
```

```

5760
```

```

5770
```

```

5780
```

```

5790
```

```

5800
```

```

5810
```

```

5820
```

```

5830
```

```

5840
```

```

5850
```

```

5860
```

```

5870
```

```

5880
```

```

5890
```

```

5900
```

```

5910
```

```

5920
```

```

5930
```

```

5940
```

```

5950
```

```

5960
```

```

5970
```

```

5980
```

```

5990
```

```

6000
```

```

6010
```

```

6020
```

```

6030
```

```

6040
```

```

6050
```

```

6060
```

```

6070
```

```

6080
```

```

6090
```

```

6100
```

```

6110
```

```

6120
```

```

6130
```

```

6140
```

```

6150
```

```

6160
```

```

6170
```

```

6180
```

```

6190
```

```

6200
```

```

6210
```

```

6220
```

```

6230
```

```

6240
```

```

6250
```

```

6260
```

```

6270
```

```

6280
```

```

6290
```

```

6300
```

```

6310
```

```

6320
```

```

6330
```

```

6340
```

```

6350
```

```

6360
```

```

6370
```

```

6380
```

```

6390
```

```

6400
```

```

6410
```

```

6420
```

```

6430
```

```

6440
```

```

6450
```

```

6460
```

```

6470
```

```

6480
```

```

6490
```

```

6500
```

```

6510
```

```

6520
```

```

6530
```

```

6540
```

```

6550
```

```

6560
```

```

6570
```

```

6580
```

```

6590
```

```

6600
```

```

6610
```

```

6620
```

```

6630
```

```

6640
```

```

6650
```

```

6660
```

```

6670
```

```

6680
```

```

6690
```

```

6700
```

```

6710
```

```

6720
```

```

6730
```

```

6740
```

```

6750
```

```

6760
```

```

6770
```

```

6780
```

```

6790
```

```

6800
```

```

6810
```

```

6820
```

```

6830
```

```

6840
```

```

6850
```

```

6860
```

```

6870
```

```

6880
```

```

6890
```

```

6900
```

```

6910
```

```

6920
```

```

6930
```

```

6940
```

```

6950
```

```

6960
```

```

6970
```

```

6980
```

```

6990
```

```

7000
```

```

7010
```

```

7020
```

```

7030
```

```

7040
```

```

7050
```

```

7060
```

```

7070
```

```

7080
```

```

7090
```

```

7100
```

```

7110
```

```

7120
```

```

7130
```

```

7140
```

```

7150
```

```

7160
```

```

7170
```

```

7180
```

```

7190
```

```

7200
```

```

7210
```

```

7220
```

```

7230
```

```

7240
```

```

7250
```

```

7260
```

```

7270
```

```

7280
```

```

7290
```

```

7300
```

```

7310
```

```

7320
```

```

7330
```

```

7340
```

```

7350
```

```

7360
```

```

7370
```

```

7380
```

```

7390
```

```

7400
```

```

7410
```

```

7420
```

```

7430
```

```

7440
```

```

7450
```

```

7460
```

```

7470
```

```

7480
```

```

7490
```

```

7500
```

```

7510
```

```

7520
```

```

7530
```

```

7540
```

```

7550
```

```

7560
```

```

7570
```

```

7580
```

```

7590
```

```

7600
```

```

7610
```

```

7620
```

```

7630
```

```

7640
```

```

7650
```

```

7660
```

```

7670
```

```

7680
```

```

7690
```

```

7700
```

```

7710
```

```

7720
```

```

7730
```

```

7740
```

```

7750
```

```

7760
```

```

7770
```

```

7780
```

```

7790
```

```

7800
```

```

7810
```

```

7820
```

```

7830
```

```

7840
```

```

7850
```

```

7860
```

```

7870
```

```

7880
```

```

7890
```

```

7900
```

```

7910
```

```

7920
```

```

7930
```

```

7940
```

```

7950
```

```

7960
```

```

7970
```

```

7980
```

```

7990
```

```

8000
```

```

8010
```

```

8020
```

```

8030
```

```

8040
```

```

8050
```

```

8060
```

```

8070
```

```

8080
```

```

8090
```

```

8100
```

```

8110
```

```

8120
```

```

8130
```

```

8140
```

```

8150
```

```

8160
```

```

8170
```

```

8180
```

```

8190
```

```

8200
```

```

8210
```

```

8220
```

```

8230
```

```

8240
```

```

8250
```

```

8260
```

```

8270
```

```

8280
```

```

8290
```

```

8300
```

```

8310
```

```

8320
```

```

8330
```

```

8340
```

```

8350
```

```

8360
```

```

8370
```

```

8380
```

```

8390
```

```

8400
```

```

8410
```

```

8420
```

```

8430
```

```

8440
```

```

8450
```

```

8460
```

```

8470
```

```

8480
```

```

8490
```

```

8500
```

```

8510
```

```

8520
```

```

8530
```

```

8540
```

```

8550
```

```

8560
```

```

8570
```

```

8580
```

```

8590
```

```

8600
```

```

8610
```

```

8620
```

```

8630
```

```

8640
```

```

8650
```

```

8660
```

```

8670
```

```

8680
```

```

8690
```

```

8700
```

```

8710
```

```

8720
```

```

8730
```

```

8740
```

```

8750
```

```

8760
```

```

8770
```

```

8780
```

```

8790
```

```

8800
```

```

8810
```

```

8820
```

```

8830
```

```

8840
```

```

8850
```

```

8860
```

```

8870
```

```

8880
```

```

8890
```

```

8900
```

```

8910
```

```

8920
```

```

8930
```

```

8940
```

```

8950
```

```

8960
```

```

8970
```

```

8980
```

```

8990
```

```

9000
```

```

9010
```

```

9020
```

```

9030
```

```

9040
```

```

9050
```

```

9060
```

```

9070
```

```

9080
```

```

9090
```

```

9100
```

```

9110
```

```

9120
```

```

9130
```

```

9140
```

```

9150
```

```

9160
```

```

9170
```

```

9180
```

```

9190
```

```

9200
```

```

9210
```

```

9220
```

```

9230
```

```

9240
```

```

9250
```

```

9260
```

```

9270
```

```

9280
```

```

9290
```

```

9300
```

```

9310
```

```

9320
```

```

9330
```

```

9340
```

```

9350
```

```

9360
```

```

9370
```

```

9380
```

```

9390
```

```

9400
```

```

9410
```

```

9420
```

```

9430
```

```

9440
```

```

9450
```

```

9460
```

```

9470
```

```

9480
```

```

9490
```

```

9500
```

```

9510
```

```

9520
```

```

9530
```

```

9540
```

```

9550
```

```

9560
```

```

9570
```

```

9580
```

```

9590
```

```

9600
```

```

9610
```

```

9620
```

```

9630
```

```

9640
```

```

9650
```

```

9660
```

```

9670
```

```

9680
```

```

9690
```

```

9700
```

```

9710
```

```

9720
```

```

9730
```

```

9740
```

```

9750
```

```

9760
```

```

9770
```

```

9780
```

```

9790
```

```

9800
```

```

9810
```

```

9820
```

```

9830
```

```

9840
```

```

9850
```

```

9860
```

```

9870
```

```

9880
```

```

9890
```

```

9900
```

```

9910
```

```

9920
```

```

9930
```

```

9940
```

```

9950
```

```

9960
```

```

9970
```

```

9980
```

```

9990
```



```

CC 1C 11=4,IX1
I=11
IF(X.LT.C(I)) GO TO 11
CONTINUE
10 IO=I-2
IX2=IX1+4
IX3=IX1+N2
DO 2C JJ=IX2,IX3
J=JJ
IF(Y.LT.C(J)) GO TO 21
CONTINUE
20 JO=J-IX1
NA=IX3+N2*((L-1)*N1+IC-2)+JO-1
F1=(X-C(I-1))/(O(I)-C(I-1))
F2=(Y-C(J-1))/(O(J)-C(J-1))
F3=F1*F2
D= C(NA)*(1.-F1-F2+F3)
1 + C(NA+N2)*(F1-F3)
2 + C(NA+1)*(F2-F3)
3 + C(NA+N2+1)*F3
RETURN
END

```

CCCC CCCCC

FUNCTION THREDL (X,Y,Z,AX,AY,AZ,XYZ,NX,NY,NZ)

THIS FUNCTION INTERPOLATES LINEARLY IN THREE DIMENSIONS. IT
 SELECTS THE TWO PLANES OF CONSTANT Z AND USES STDLI FOR TWO DIMEN-
 SIONAL INTERPOLATION. A LINEAR INTERPOLATION IS USED IN Z.
 PROGRAMMER H. SOBEL
 DATE 22 AUGUST 1968
 DIMENSION AX(NX), AY(NY), AZ(NZ), XYZ(NX,NY,NZ)
 DATA N/1/
 CALL SEFCF (Z,AZ,K,NZ,M)
 AL=STDLIA(AX,AY,XYZ(1,1,K),X,Y,NX,NY)
 AU=STDLIA(AX,AY,XYZ(1,1,K+1),X,Y,NX,NY)
 THRECL=AL+(AU-AL)*(Z-AZ(K))/(AZ(K+1)-AZ(K))
 RETURN
 END

F 2
 F 4
 F 6
 F 8
 F 10
 F 12
 F 14
 F 16
 F 18
 F 20
 F 22
 F 24
 F 26-

LIST OF REFERENCES

1. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 9-17, 24 March 1978.
2. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 11, 24 March 1978.
3. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 21, 24 March 1978.
4. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 1-1.
5. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and others, p. C-1-C-37.
6. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and others, p. C-38-C-51.
7. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-18.
8. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-16.
9. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-37.
10. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-10.
11. Naval Postgraduate School Technical Note 0141-25, User's Guide to Programming Aids Library, by Lloyd G. Nolan, p. 27-29, October 1978.
12. Honeywell Defense Systems Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, 24 March 1978.

13. Honeywell Defense Systems Division, Systems Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 77-82, 24 March 1978.
14. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MCD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 4-41-4-83.
15. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and Others, p. B-1 - B-49.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
2. Department Chairman, Code 67 Department of Aeronautical Engineering Naval Postgraduate School Monterey, California 93940	1
3. Professor D. J. Collins, Code 67C0 (thesis advisor) Department of Aeronautical Engineering Naval Postgraduate School Monterey, California 93940	2
4. LT Thomas Alan Grote, USN (student) 1016 Orchid Avenue Modesto, California 95355	1
5. Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2

INITIAL DISTRIBUTION LIST

No. Copies

1. Library, Code 0102
Naval Postgraduate School
Monterey, California 93940

2. Department of Aeronautical Engineering
Naval Postgraduate School
Monterey, California 93940

3. Professor D. J. Collins, Code 0102 (Special Advisor)
Department of Aeronautical Engineering
Naval Postgraduate School
Monterey, California 93940

4. Lt Thomas Alan Butler, USN (Student)
1001 Grand Avenue
Modesto, California 95350

5. Defense Technical Information Center
Cameron Station
Alexandria, Virginia 22304

Thesis
G86075
c.1

Grote

Guidance and control
of tactical missiles.

20 APR 83
20 APR 83
OCT 3 85

188001

29416
53039

8801

control
files.

39

Thesis
G86075
c.1

Grote

Guidance and control
of tactical missiles.

188001

thesG86075

Guidance and control of tactical missile



3 2768 002 13568 3

DUDLEY KNOX LIBRARY